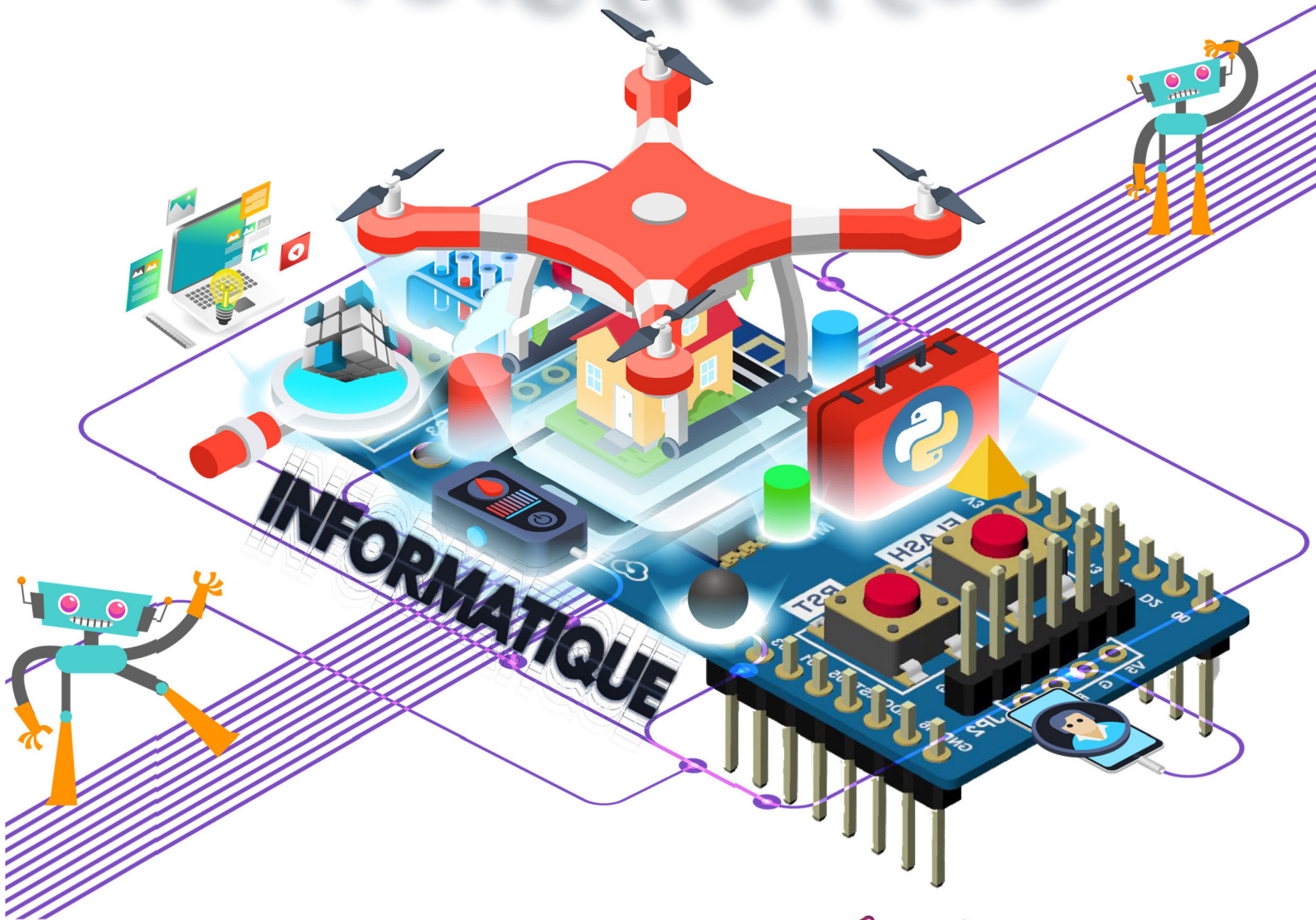


# MON CAHIER d'activités



1ÈRE ANNÉE  
SECONDAIRE

# RÉALISATEURS

**Sihem Midouni**

ENSEIGNANTE ÉMÉRITE HORS  
CLASSE

**Nadia El Mekni**

ENSEIGNANTE ÉMÉRITE HORS  
CLASSE

**Ibtissem Ben Arbia**

ENSEIGNANTE ÉMÉRITE HORS  
CLASSE

**Youssef Douira**

ENSEIGNANT ÉMÉRITE

**Aymen Gharbi**

ENSEIGNANT ÉMÉRITE

**Chiheb Akili**

ENSEIGNANT ÉMÉRITE

**Abdelhalim Makhzoumi**

ENSEIGNANT ÉMÉRITE HORS CLASSE

**Houcine Guediri**

ENSEIGNANT PRINCIPAL HORS CLASSE

**Encadré et évalué par :**

**Besma Tounsi Hmaidi**

INSPECTRICE GÉNÉRALE

1ERE ANNÉE  
SECONDAIRE



# Sommaire



## 01 PRESENTATION GENERALE



## 02 PRODUCTION NUMERIQUE



## 03 PROGRAMMATION



## 04 ROBOTIQUE





# PROJET

## Smart Cross Road



**Smart Cross-Road** ou **croisement intelligent**, est équipé d'outils technologiques qui interagissent avec l'environnement (usagers et nature) pour permettre un transport sûr, flexible et durable.

Ils combinent une infrastructure physique (telle que des capteurs et feux intelligents) avec une infrastructure logicielle (telle que l'IA "**Intelligence Artificielle**" et les données).

Le but de ce projet est d'intégrer les compétences et les connaissances techniques acquises tout au long de l'année grâce à un travail effectué en équipe et d'y ajouter un esprit de challenge et de créativité.

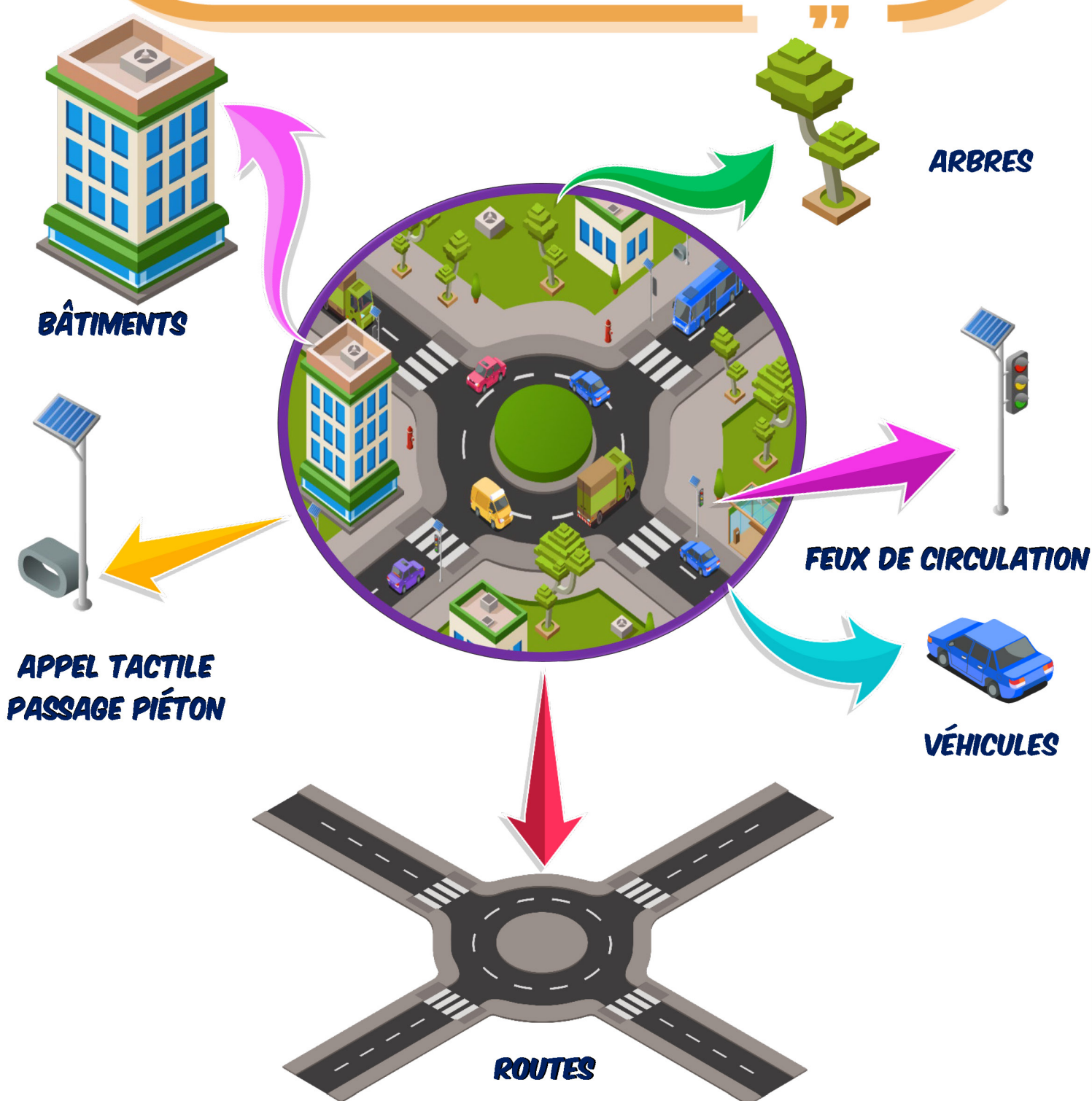


# DÉCOMPOSITION DU PROJET

Votre future conception du SMART CROSS ROAD contiendra plusieurs objets qui vont être modélisés en utilisant un outil de modélisation 3D, et certains de ces objets doivent être programmés afin de rendre votre carrefour intelligent, ce qui vous fera plonger dans l'univers de la robotique en développant vos propres cartes.

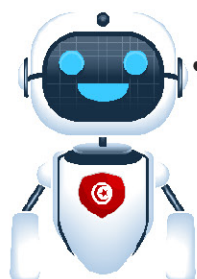
Ainsi vous découvrirez le monde de programmation utilisant le langage Python.

Ci-dessous un exemple de décomposition d'un croquis représentant un Smart cross Road.



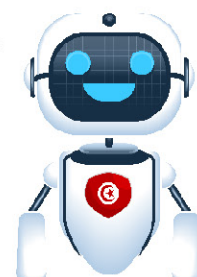
**VOTRE  
NOM ET PRÉNOM ?**

1S ..... GROUPE .....



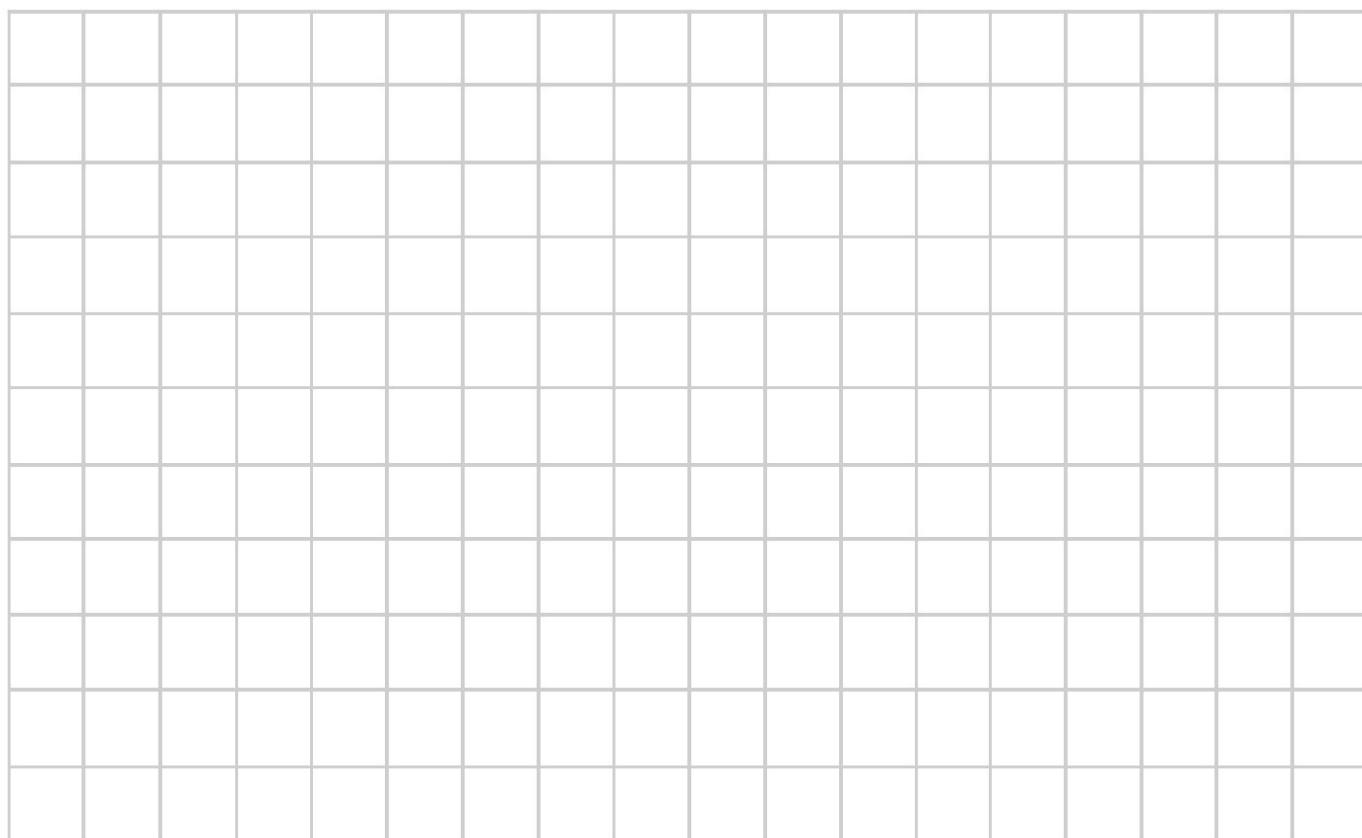
**VOTRE RÔLE AU SEIN DU GROUPE ?**

**LES NOMS DU RESTE DE L'ÉQUIPE ?**



**DESSINER LE CROQUIS DE  
VOTRE FUTUR SMART  
CROSS ROAD**

# Mon Croquis





# Production Numérique

## 2D vers 3D



Observer attentivement  
cette image.



Citer les  
Les objets qui  
figurent dans cette  
image.



Classer les objets 2D et 3D dans le tableau ci-dessous.

OBJET 3D

|       |
|-------|
| ..... |
| ..... |
| ..... |
| ..... |
| ..... |
| ..... |



OBJET 2D

|       |
|-------|
| ..... |
| ..... |
| ..... |
| ..... |
| ..... |
| ..... |



Scanner le QR code afin de visualiser la capsule et compléter le paragraphe par les mots à partir de la liste pour définir la notion 2D.

LISTE

LA LARGEUR Y, LA LONGUEUR X, AUDIOVISUEL  
DEUX DIMENSIONS, PLANE, DESSIN, PEINTURE



LA 2D, QU'EST-CE QUE C'EST ?

La 2D signifie .....

La présentation d'une forme en 2D est définie par deux données :

et .....

Ainsi une forme en 2D est une forme .....

La 2D est utilisée dans divers domaines comme : .....

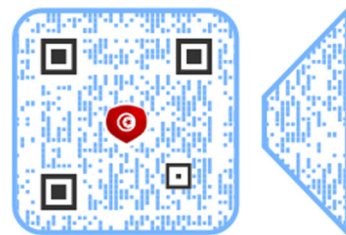




Scanner le QR code et visualiser la capsule afin de compléter le paragraphe par les mots à partir de la liste pour définir la notion **3D**.

**LISTE**

**LA HAUTEUR, L'IMPRESSION, VOLUME,  
TROIS DIMENSIONS, LES BÂTIMENTS**



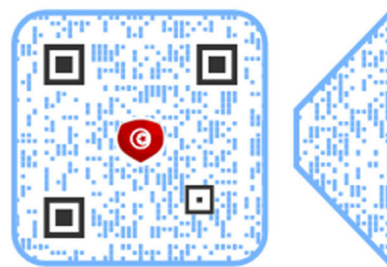
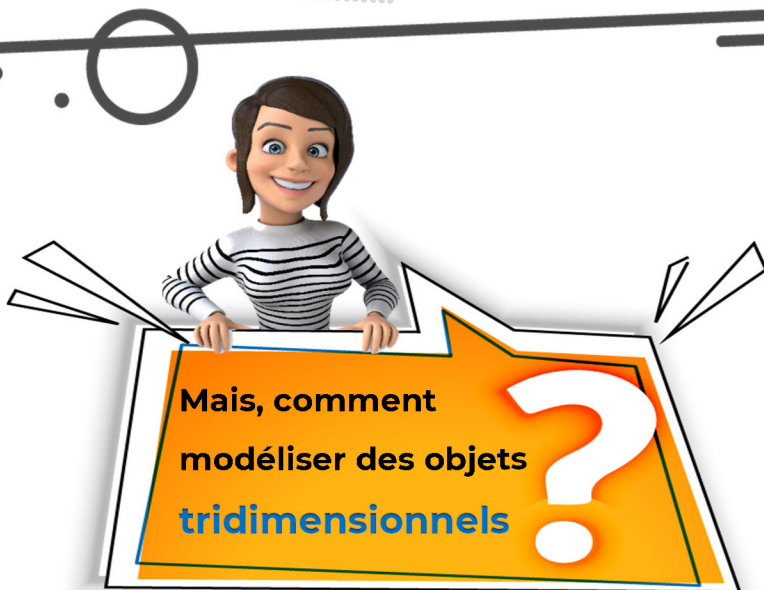
## **QU'EST-CE QUE LA 3D ?**

La **3D** signifie .....

Avec la **3D**, une troisième donnée est ajoutée : c'est .....

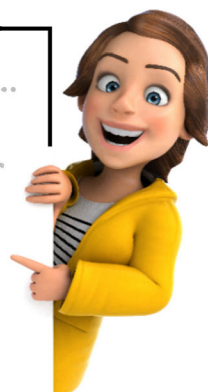
La modélisation en **3D** donne du ..... à l'objet présenté, afin de montrer chaque détail de ce dernier .

La **3D** est utilisée dans divers domaines comme : .....  
et .....



Il faut tout d'abord avoir un ..... de .....  
installé dans l'ordinateur, permettant de créer des .....  
en 3D , dans un espace .....

Ces logiciels sont nombreux, certains sont gratuits d'autres payants, et chacun possède ses propres points forts.





**Mener une recherche sur les différents logiciels de modélisation d'objets 3D et compléter les fiches des logiciels suivants :**



Nom : .....

Licence : .....

Domaine d'utilisation : .....

Nom : .....

Licence : .....

Domaine d'utilisation : .....



Nom : .....

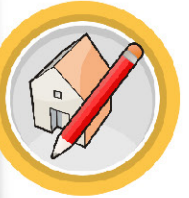
Licence : .....

Domaine d'utilisation : .....

Nom : .....

Licence : .....

Domaine d'utilisation : .....



Hé, pour bien débiter notre apprentissage dans l'univers de la **3D** on utilisera **Google SketchUp** et cela pour trois bonnes raisons :

**Google SketchUp** est sous licence gratuite,

il possède une interface peu complexe et facile à exploiter,

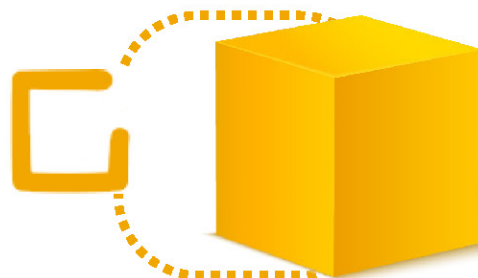
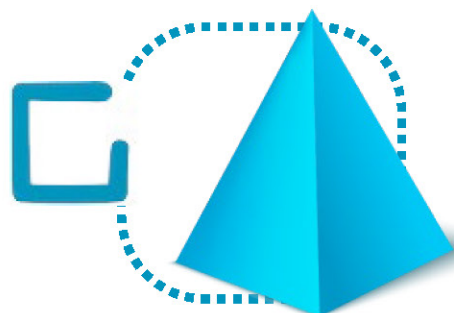
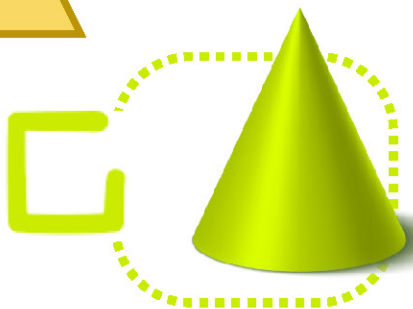
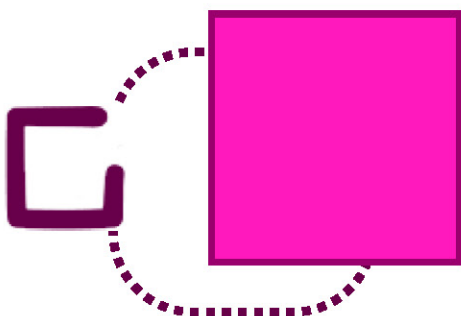
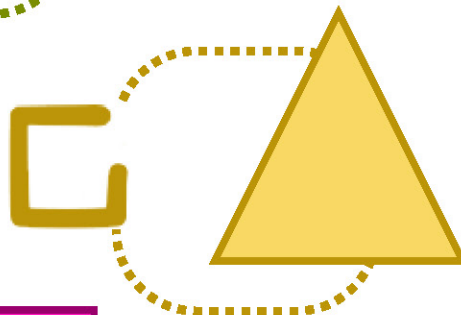
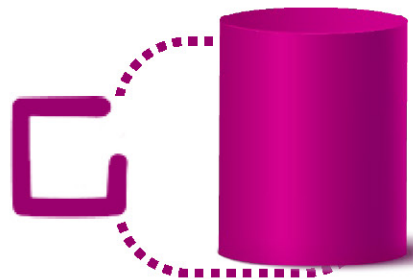
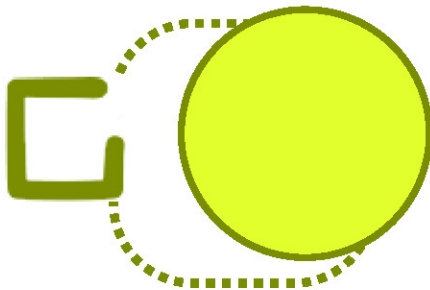
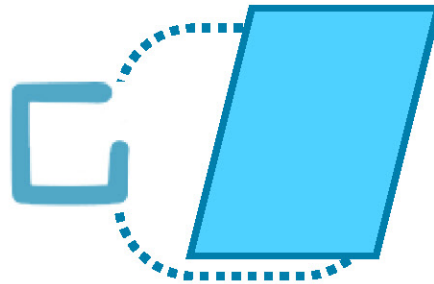
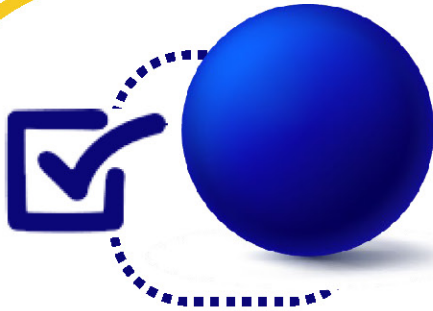
il est polyvalent, et peut être utilisé dans de nombreux secteurs.

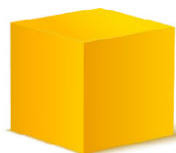




**AIDEZ- MOI À IDENTIFIER LES FORMES 3D !**

Cocher  chaque image en dessous qui représente la forme d'un **objet 3D**.





# Création des Formes Simples

Avec Google SketchUp



Avant la modélisation des **objets 3D** avec SketchUp,  
il faut paramétrer le logiciel.  
En effet il faut choisir **l'unité de mesure**,  
afficher **le grand jeu d'outils** etc....



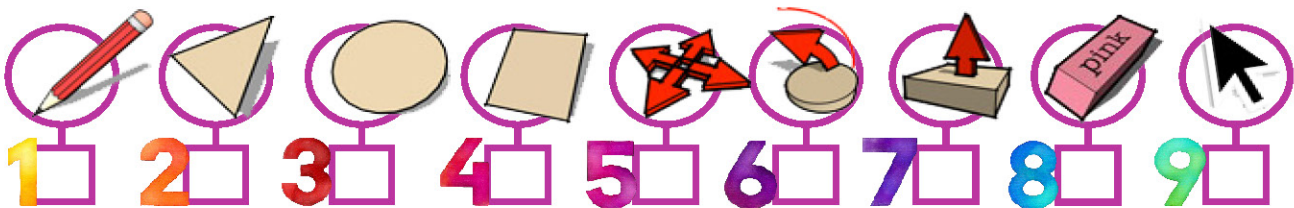
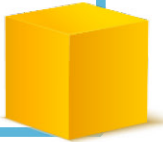


# Sur machine



Je coche les outils nécessaires pour modéliser un cube.

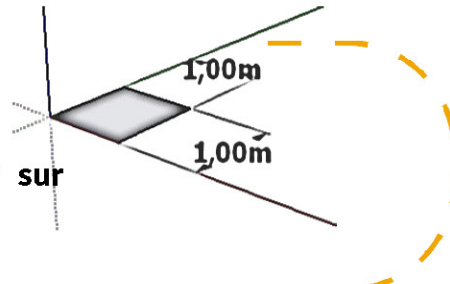
Je complète les étapes par le numéro de l'outil, son nom ou son rôle.



Je dessine un carré en utilisant l'outil

N° ..... nommé **RECTANGLE**.

Je place le carré dans la zone de dessin sur l'origine des trois axes.



Dimensions 1m;1m

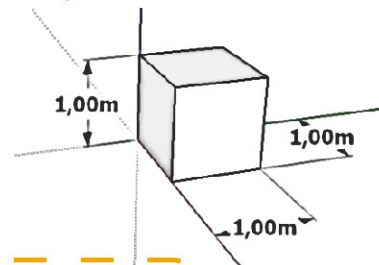
Je saisis les dimensions et je valide par la touche **ENTREE**

Les dimensions sont séparées par un ;

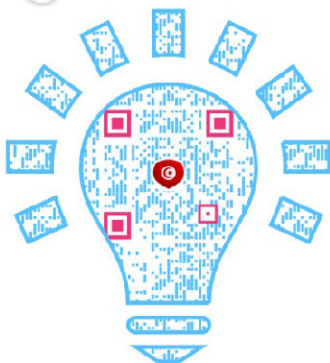
Je donne du volume au carré en utilisant l'outil

N° ..... nommé **POUSSER /TIRER**.

Je saisis la hauteur et je valide.

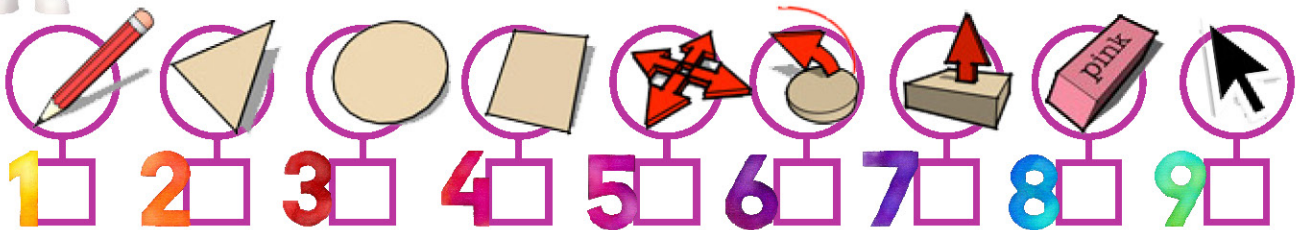


Je peux pivoter la caméra autour de l'objet grâce à l'outil **Orbite**.

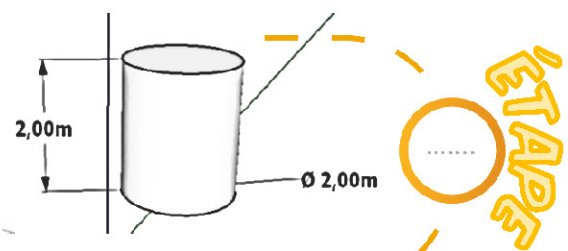




Je coche les outils nécessaires pour modéliser un cylindre.  
J'ordonne les actions ci-dessous en précisant le numéro de l'étape afin de modéliser un cylindre.

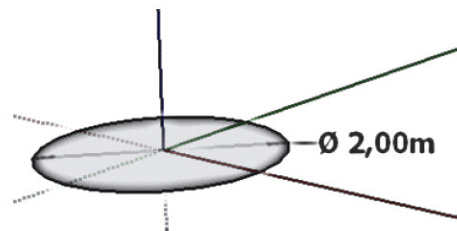



Je donne du volume au cercle dessiné en utilisant l'outil N° 7 nommé **POUSSER/ TIRER**.



ETAPE

Je dessine un cercle à partir de l'origine du repère en utilisant l'outil N° 3 nommé **CERCLE**.



Pour voir l'objet de différentes faces je peux déplacer la caméra horizontalement ou verticalement grâce à l'outil **panoramique** 

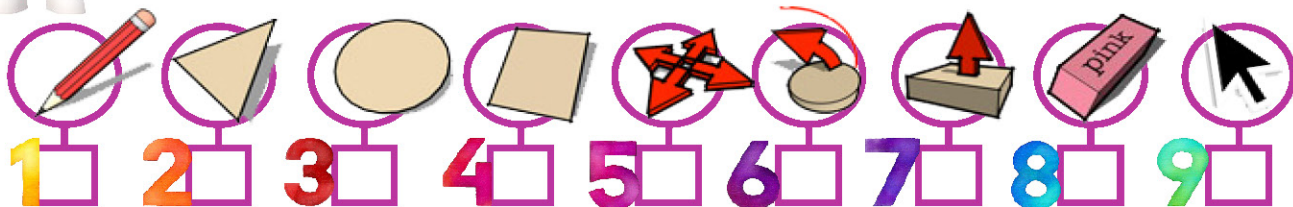






Je coche les outils nécessaires pour modéliser un cône.

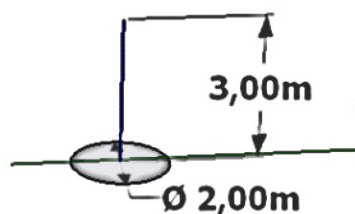
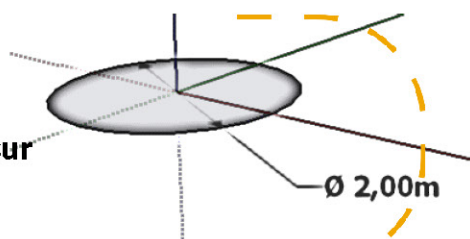
Je complète les étapes par le numéro de l'outil, son nom ou son rôle.



Je dessine un cercle en utilisant l'outil

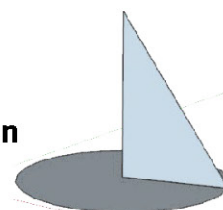
N° ..... nommé **CERCLE**.

Je place le cercle dans la zone de dessin sur l'origine des trois axes.



J'utilise l'outil N° 1 nommé **LIGNE** et je clique sur la flèche de direction « haut » du clavier pour verrouiller l'axe **Bleu**.

Avec l'outil N° ..... je dessine un triangle dans un plan perpendiculaire au plan qui contient le cercle.

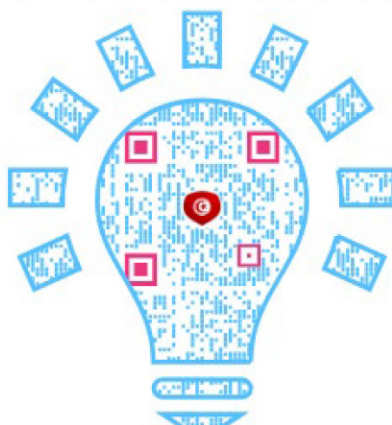


Je sélectionne le cercle avec l'outil **SÉLECTION** N° .....

J'utilise l'outil N° ..... nommé **SUIVEZ-MOI** et je clique- sur le triangle afin de .....

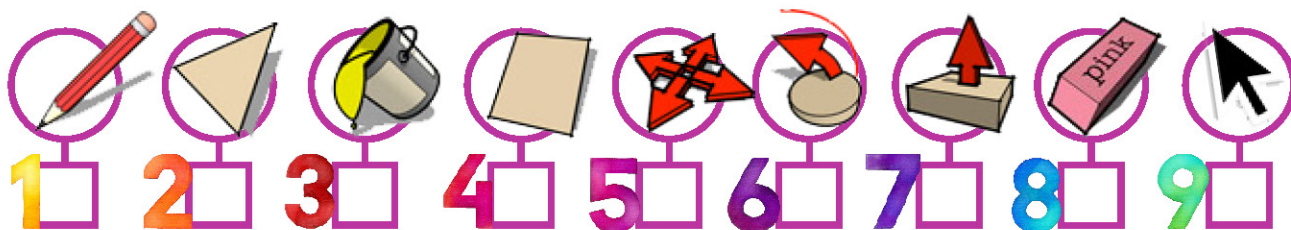
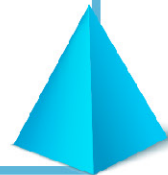
J'utilise l'outil N°8 nommé **LIGNE** afin de fermer la surface du cercle.

J'efface la ligne dessinée dans l'étape précédente.

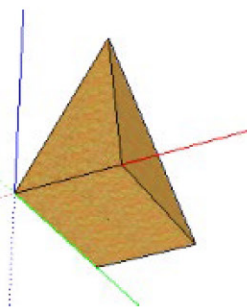




Je coche les outils nécessaires pour modéliser une pyramide.  
J'ordonne les actions ci-dessous en précisant le numéro de l'étape afin de modéliser la pyramide.



ÉTAPE



J'utilise l'outil N° 3 nommé **COLORIER** et je clique sur l'option **SÉLECTIONNER**.

Je choisis l'option **Brique et revêtement**.

Je choisis **Brique\_brute\_foncée**

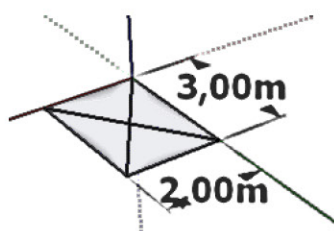
Je clique sur l'objet.

Afin de fermer le rectangle de base, je dessine une diagonale sur le rectangle de base en utilisant l'outil **LIGNE**, puis je l'efface en utilisant l'outil N° 8 nommé **EFFACER**.

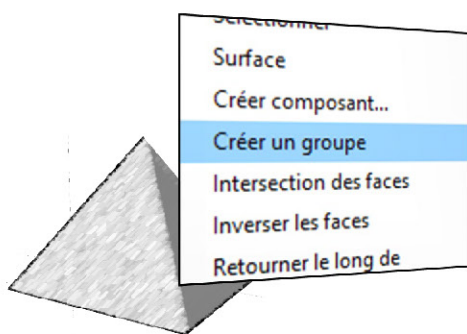


ÉTAPE

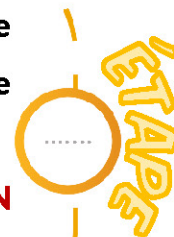
ÉTAPE



D'abord je dessine un rectangle de base, ensuite je dessine les deux diagonales du rectangle en utilisant l'outil N° 1 nommé **LIGNE**.



Dans l'objectif de transformer l'objet en une seule entité, Je clique avec le bouton droit de la souris sur l'objet puis je déroule le menu contextuel et je choisis l'option **CRÉER UN GROUPE**.

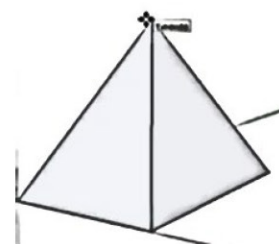


En vue de donner du volume à la pyramide, j'utilise l'outil N° 5 nommé **DÉPLACER** et je clique sur la flèche de direction haut du clavier pour verrouiller le déplacement parallèlement à l'axe bleu.



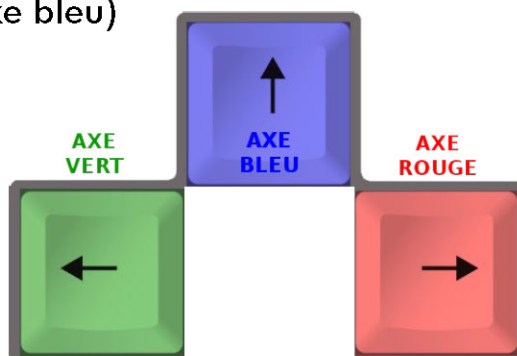
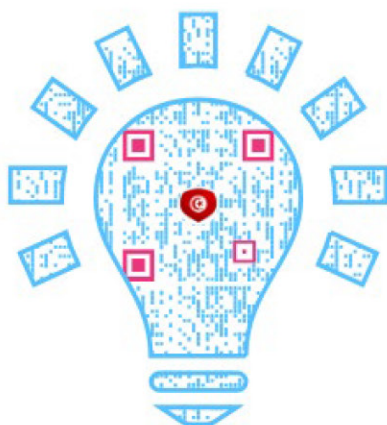
Je sélectionne le centre du rectangle (intersection de deux diagonales) et je tire vers le haut.

Je saisis la valeur de la hauteur et je valide par la touche **ENTREE**.



Lors de l'utilisation de l'outil **déplacer**, on peut bloquer le déplacement selon un axe précis.

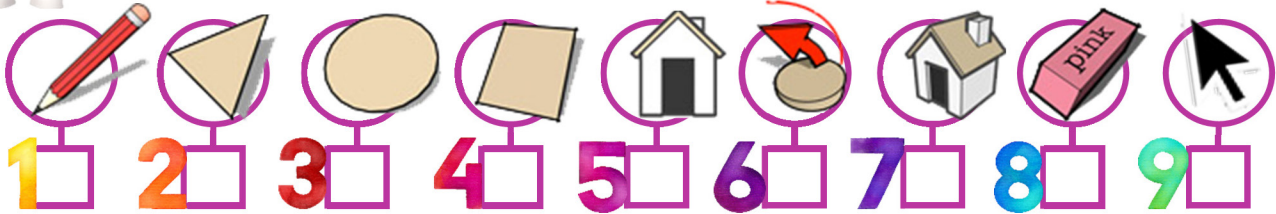
Il suffit d'appuyer une fois sur la flèche de direction droite du clavier (axe rouge), gauche (axe vert) ou verticale (axe bleu)



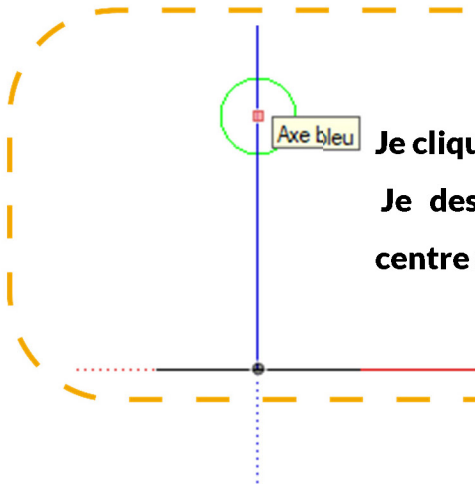
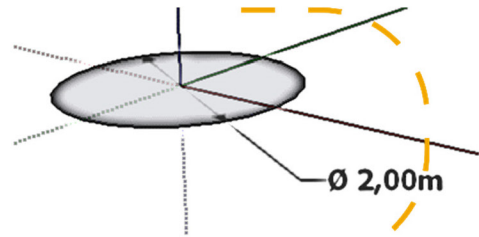




Je coche les outils nécessaires pour modéliser une sphère.  
Je complète les étapes par le numéro de l'outil, son nom  
ou son rôle.



Je dessine un cercle en utilisant l'outil  
N° 3 nommé ..... Je place le cercle dans la  
zone de dessin sur l'origine des trois axes.



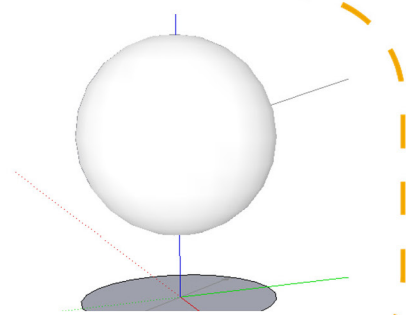
Je clique sur la vue **FACE** grâce à l'outil N° .....

Je dessine un deuxième cercle de rayon **1m** dont le  
centre se situe sur l'axe bleu.

Je sélectionne le premier cercle.

J'utilise l'outil N° ..... nommé **SUIVEZ-MOI**

Je clique sur le deuxième cercle.




J'utilise les vues pour voir l'objet de différentes faces.





# Projet Smart Cross-Road : quelques composants



Cocher  la(les) forme(s) simple(s) à utiliser pour modéliser le panneau de signalisation "STOP"



Solent les composants du panneau de signalisation "STOP", donner les dimensions standards de chaque partie

Les dimensions standards du panneau "STOP" sont :

1) Panneau

.....  
.....

2) Pied du panneau

.....  
.....



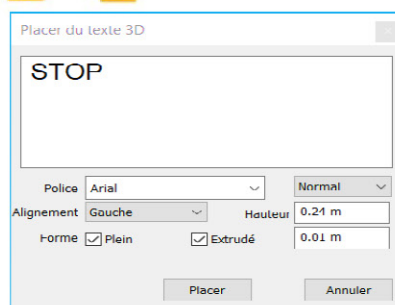
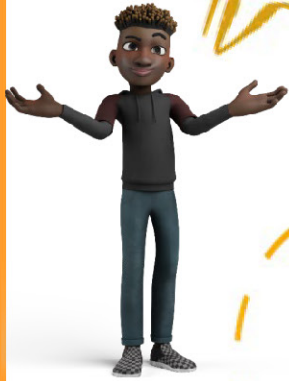
L'outil "Créer un composant"



permet de convertir un objet en un composant réutilisable

# Panneau de signalisation " STOP "

Modéliser le panneau de signalisation " **STOP** " en respectant les mesures standards



Ordonner les étapes suivantes afin de modéliser le panneau de signalisation " **STOP** "



- ☐ Colorier le contour en **blanc** et le reste en **rouge**
- ☐ Dessiner un octogone de côté **40 cm**
- ☐ Insérer le texte **STOP** sur le panneau
- ☐ Appliquer une épaisseur de **2 cm** à l'octogone
- ☐ Faire un décalage interne de **5 cm**



L'outil " **Texte** "  permet d'insérer un texte.

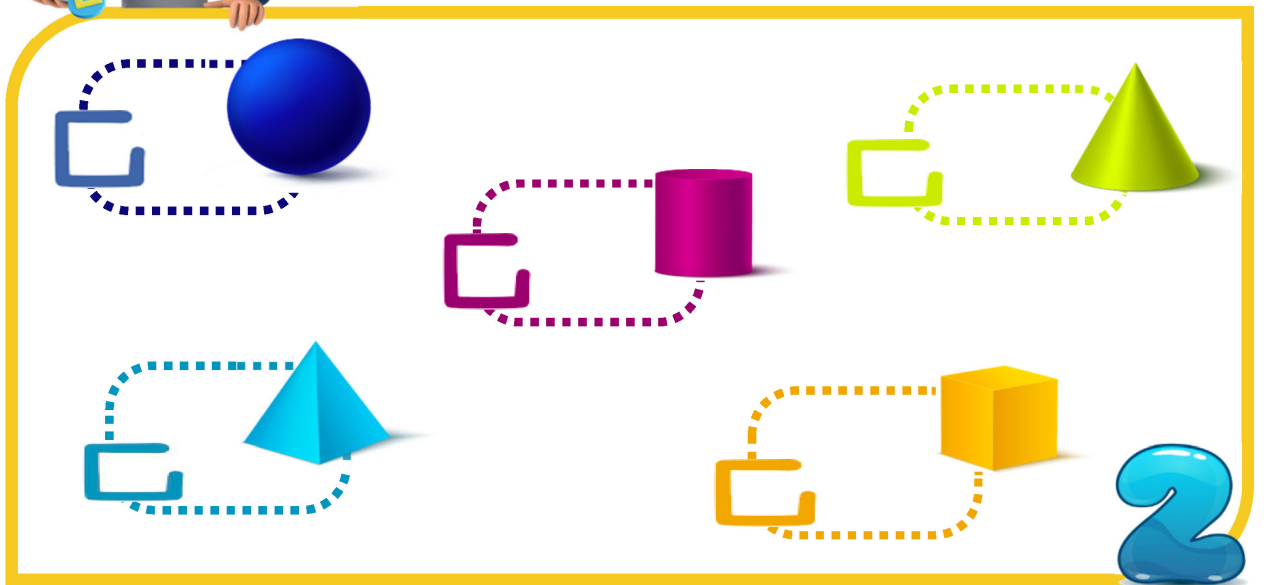
L'outil " **Mise à l'échelle** "  permet d'ajuster la taille du texte.



# Cross Road



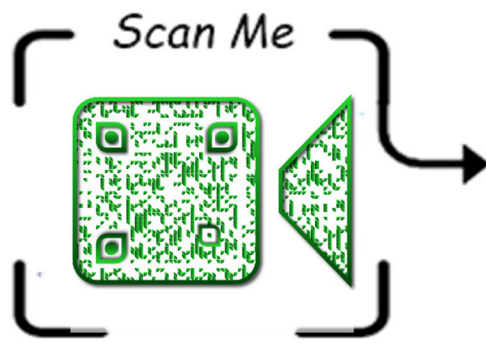
Cocher  la(les) forme(s)  
simple(s) à utiliser pour modéliser  
le **Cross Road**



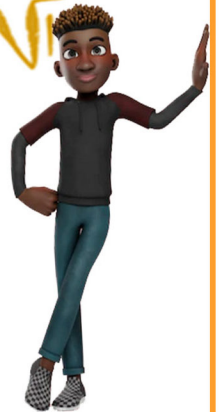
Modéliser le "**Rond-point**"  
du Smart Cross-Road selon les  
dimensions suivantes :



| Forme         | Dimensions                         |
|---------------|------------------------------------|
| Cylindre vert | Diamètre : 5m<br>Hauteur : 20 cm   |
| Cercle 1      | Diamètre : 6.5m                    |
| Cercle 2      | Diamètre : 8m                      |
| Arc de cercle | Longueur : 20cm<br>Largeur : 10 cm |

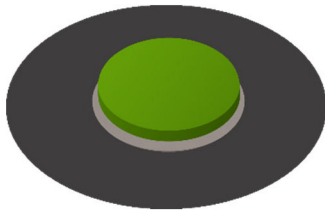
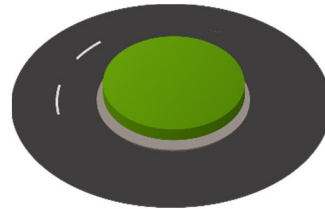


Ordonner les étapes suivantes afin de modéliser le " **Rond-point** "



ÉTAPE

Je sélectionne l'arc, tout en appuyant sur la touche CTRL je clique sur l'outil pivoter ensuite je fais une rotation.

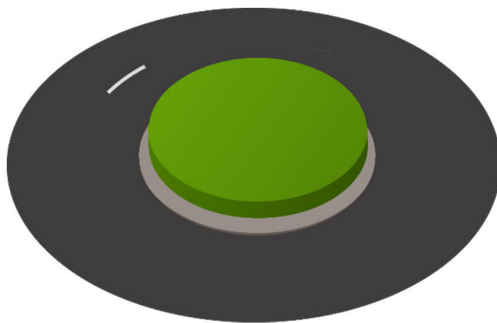


Je dessine deux cercles de même centre et de rayons différents.  
J'ajoute un cylindre vert sur le cercle le plus petit.

ÉTAPE

ÉTAPE

A la suite je saisis au clavier dans la zone cotation \*9



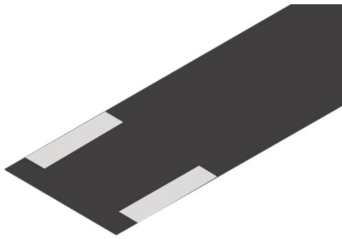
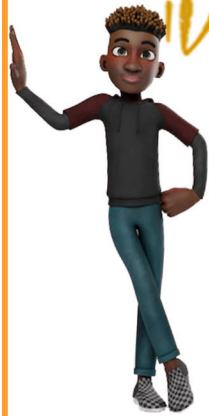
Je dessine un arc de cercle blanc.

ÉTAPE



L'outil " **Pivoter** "  permet également de créer une ou plusieurs copies de l'objet sélectionné.

Ordonner les étapes suivantes afin de modéliser le " **Passage piéton** "

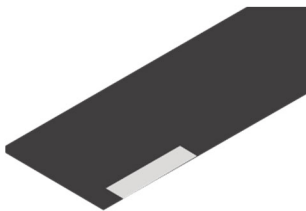
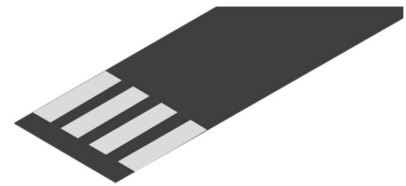
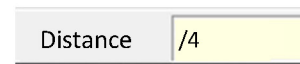


Je sélectionne le rectangle, tout en appuyant sur la touche CTRL, je clique sur l'outil déplacer ensuite je fais un déplacement vers l'autre bord de la route.

ETAPPE

ETAPPE

A la suite je saisis au clavier dans la zone cotation /4



Je dessine un rectangle blanc de longueur 3m et de largeur 0.5m.

ETAPPE



L'outil " Déplacer "




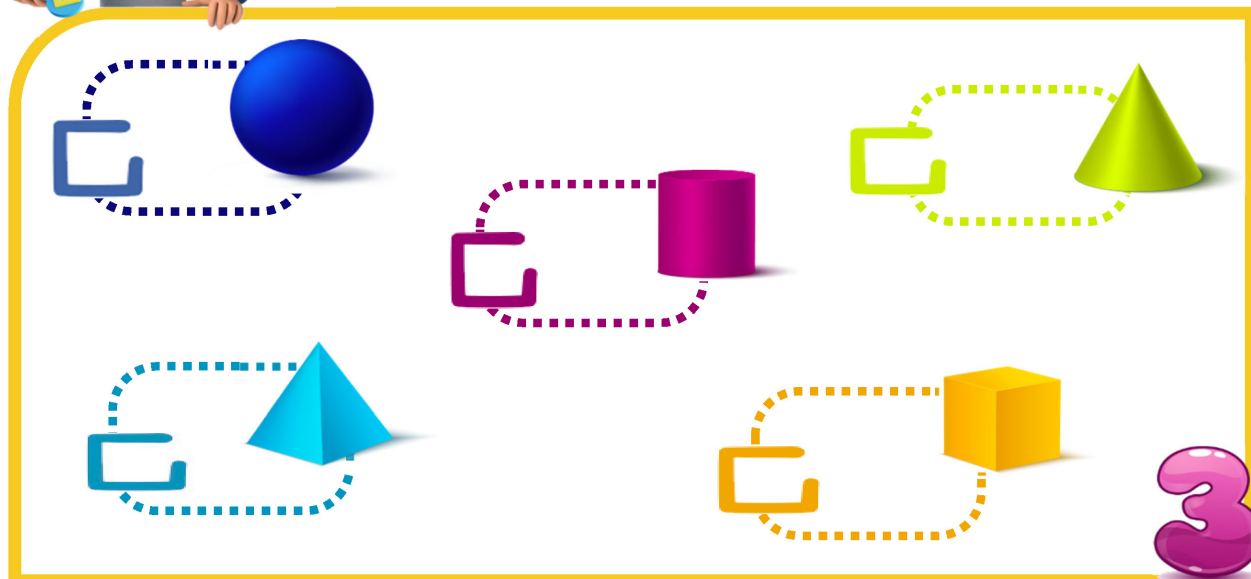
permet également de créer une ou plusieurs copies de l'objet sélectionné.



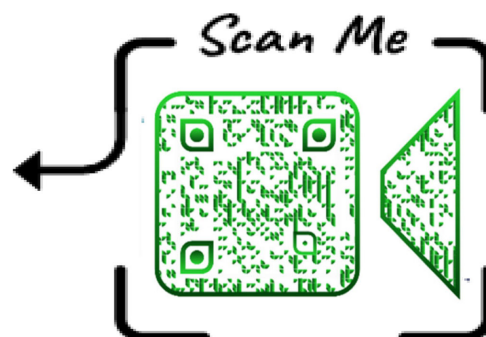
# Feu de circulation



Cocher  la(les) forme(s) simple(s) à utiliser pour modéliser le **Feu de circulation**.



Scanner le QR code ci-dessous afin de visualiser la vidéo qui explique comment modéliser le feu de circulation





*“ Un bon programmeur est quelqu'un qui regarde toujours des deux côtés avant de traverser une rue à sens unique ”*



# PyPilot

- **HEY, JE M'APPELLE "PYPILOT"**
- **JE SUIS CONÇU POUR CONDUIRE UNE VOITURE AUTONOME.**



- **MA TÂCHE CONSISTE À CONDUIRE LA VOITURE AFIN D'ATTEINDRE UN OBJECTIF**
- **CHAQUE OBJECTIF EST DÉCRIT PAR UNE MISSION.**

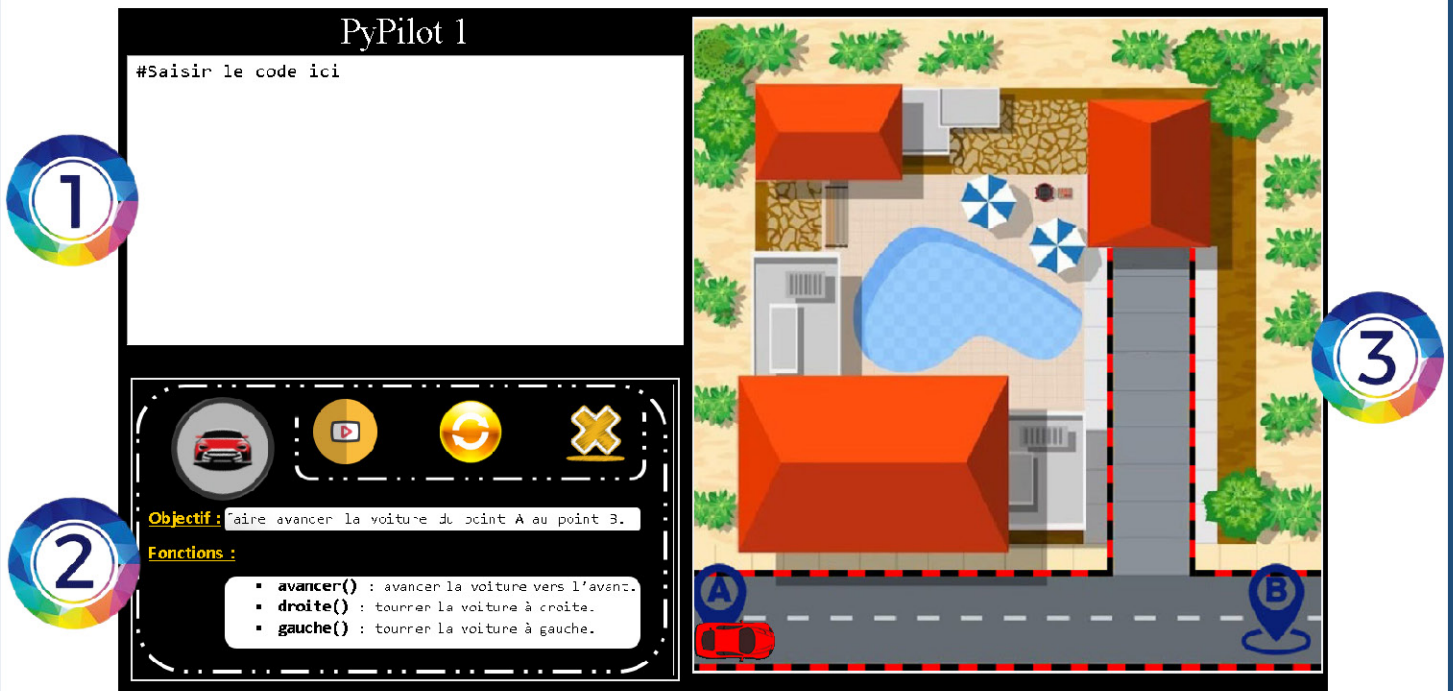


- **J'EXÉCUTE DES INSTRUCTIONS POUR ACCOMPLIR UNE MISSION**
- **LES INSTRUCTIONS SONT IMPLÉMENTÉES DANS UN PROGRAMME**
- **LES PROGRAMMES SONT ÉCRITS EN PYTHON**

**CHAQUE PROGRAMME D'UNE MISSION EST COMPOSÉ PAR :**

- **DES STRUCTURES (SIMPLES, ITÉRATIVES ET CONDITIONNELLES)**
- **DES VARIABLES**
- **DES MODULES**

# Interface



**BOUTONS DE  
CONTRÔLES**



**1** C'est la zone de saisie du programme de la mission

**2** Description des objectifs et des fonctions de contrôle du programme

**3** Map d'exécution du programme



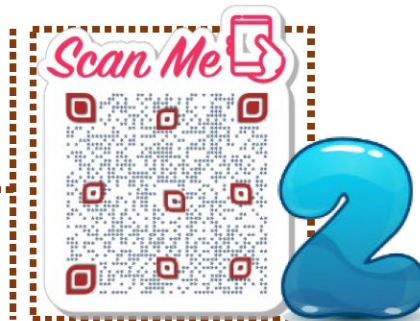
10



“ LANCER LE PROGRAMME **PYPILOT1** ET EXÉCUTER LE CODE.

*QUE PEUT-ON CONSTATER ?* ”

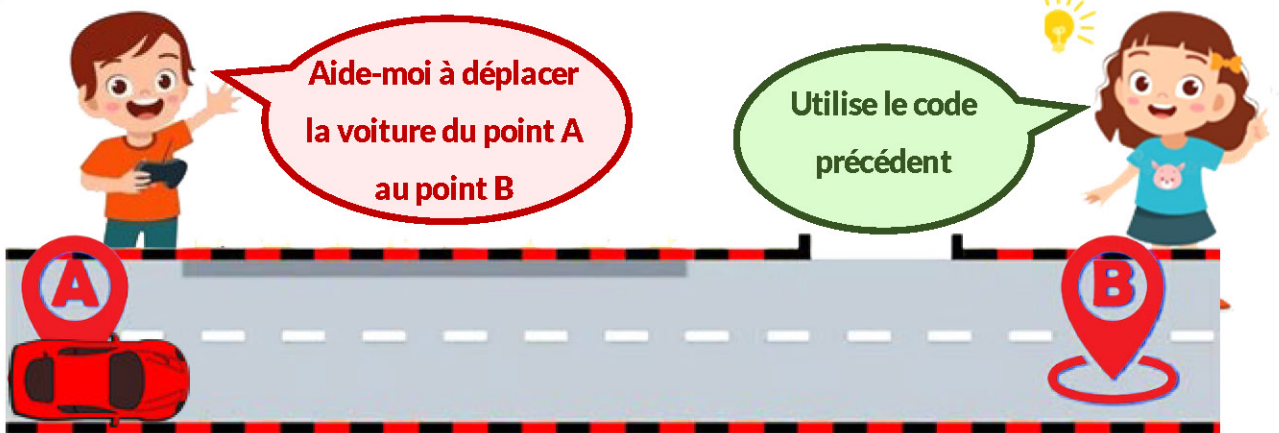
Je constate :



**for** ..... **in range** ( ..... ) :

Traitement à répéter





## OBJECTIF

Se déplacer du point **A** vers le point **B**

## FONCTIONS DE CONTRÔLE

**avancer()** : déplacer la voiture vers l'avant  
d'une unité (chaque unité est représentée  
par 4 blocs)



## CONTRAINTES

Réécrire le code en deux lignes

“ LANCER LE PROGRAMME **PYPILOT1** ET MODIFIER  
LE CODE AFIN D'ATTEINDRE L'OBJECTIF DÉCRIT  
PRÉCÉDEMMENT EN UTILISANT LA FONCTION DE  
CONTRÔLE ADÉQUATE ET EN RESPECTANT LA  
CONTRAINTÉ ASSOCIÉE. ”



Ecrire  
le  
Code



.....

.....

.....

.....

.....

.....

itération

boucle

compteur

range

Indentation



**NB :** .....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....



# PyPilot2



## OBJECTIF

Se déplacer du point A vers le point B

## FONCTIONS DE CONTRÔLE

**avancer()** : déplacer la voiture vers l'avant

**droite()** : tourner à droite de 90°

**gauche()** : tourner à gauche de 90°

## CONTRAINTES

Le code ne doit pas dépasser 5 lignes

“

LANCER LE PROGRAMME **PYPILOT2** ET TAPER LE  
CODE PERMETTANT D'ATTEINDRE L'OBJECTIF  
DÉCRIT PRÉCÉDEMMENT EN UTILISANT LES  
FONCTIONS DE CONTRÔLE ADÉQUATES EN  
RESPECTANT LA CONTRAINTE ASSOCIÉE.

”



.....

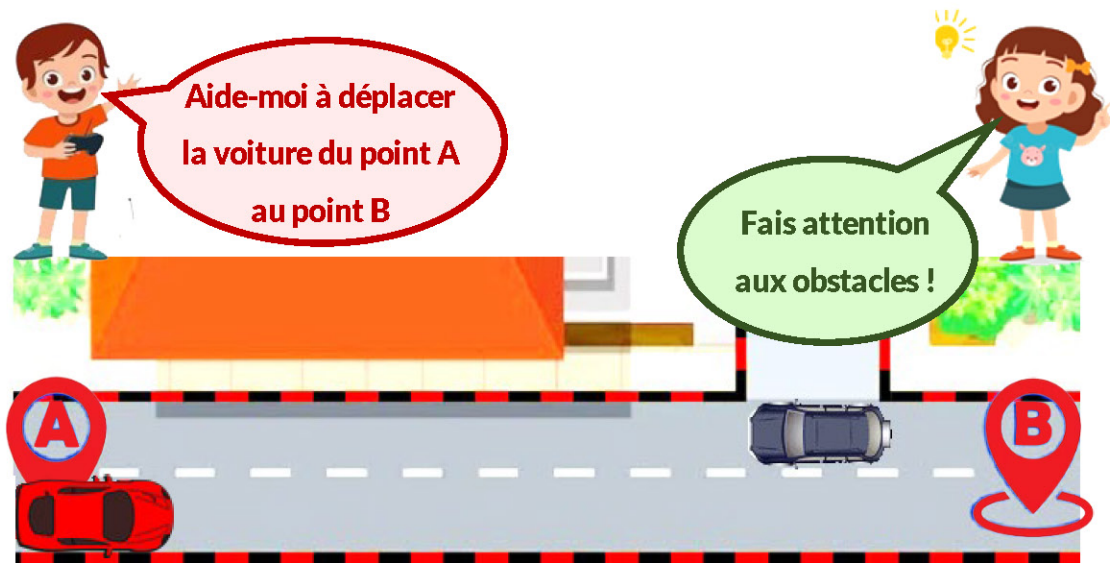
.....

.....

.....

.....

.....



## OBJECTIF

- Se déplacer du point A vers le point B
- S'arrêter s'il existe un obstacle

## FONCTIONS DE CONTRÔLE

- `avancer()` : déplacer la voiture vers l'avant
- `droite()` : tourner à droite de 90°
- `gauche()` : tourner à gauche de 90°
- `detecter()` : retourne **vrai** s'il y a un obstacle

## CONTRAINTES

**Le code ne doit pas dépasser 5 lignes**



“ LANCER LE PROGRAMME **PYPILOT3** ET TAPER LE CODE PERMETTANT D'ATTEINDRE L'OBJECTIF DÉCRIT PRÉCÉDEMMENT EN UTILISANT LES FONCTIONS DE CONTRÔLE ADÉQUATES EN RESPECTANT LA CONTRAINTE ASSOCIÉE. ”



Est-ce que je peux utiliser la boucle for ?

1



**While** ( ) :

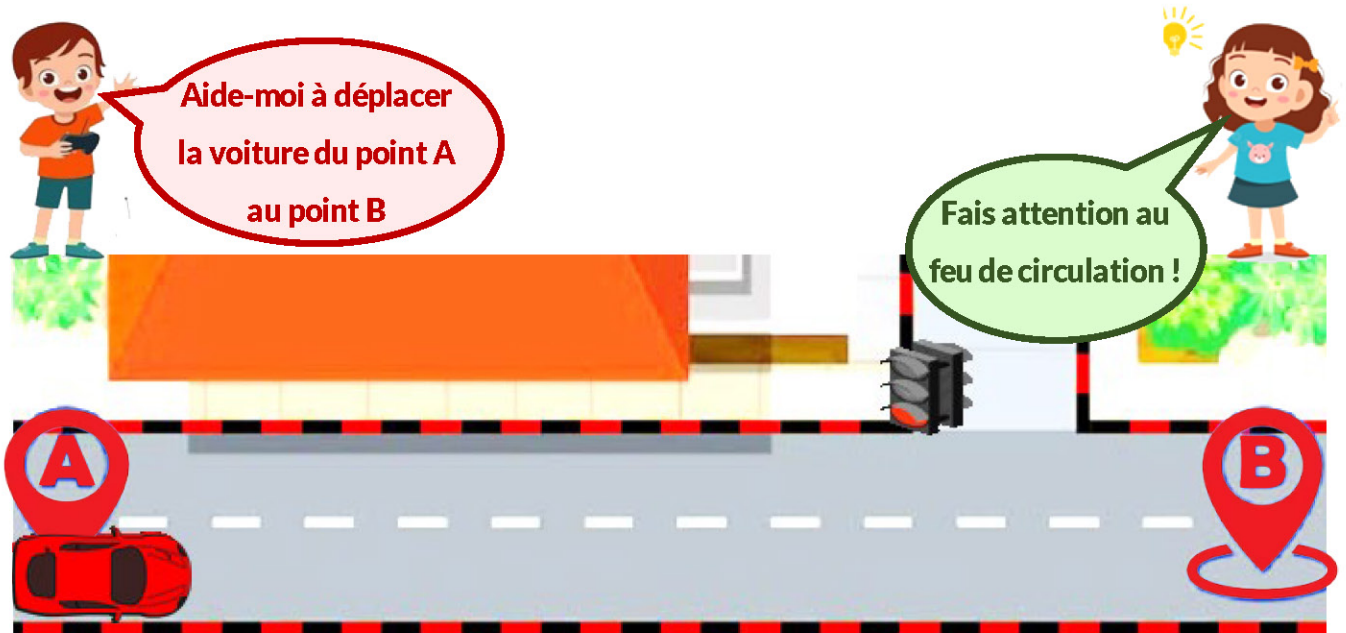
Traitement à répéter

3

Ecrire  
le  
Code



# PyPilot4



## OBJECTIF

- Passer au feu vert
- Ralentir au feu orangé
- S'arrêter au feu rouge

## FONCTIONS DE CONTRÔLE

- `avancer()` - `droite()` - `gauche()`
- `detecter_feu()` : retourne la couleur du feu  
"R", "V" ou "O"
- `freiner()` : arrêter la voiture
- `ralentir()` : ralentir la vitesse de la voiture

## CONTRAINTES

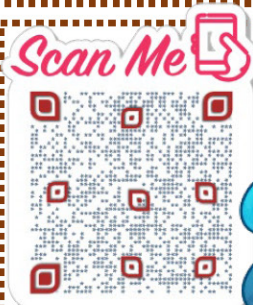
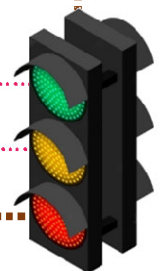
- Compléter le code troué



“ LANCER LE PROGRAMME **PYPILOT4** ET TAPER LE CODE PERMETTANT D'ATTEINDRE L'OBJECTIF DÉCRIT PRÉCÉDEMMENT EN UTILISANT LES FONCTIONS DE CONTRÔLE ADÉQUATES EN RESPECTANT LA CONTRAINTE ASSOCIÉE. ”



Selon l'état du feu, Je devrais utiliser la structure .....



```
if ( ..... ) :  
    Traitement 1  
elif ( ..... ) :  
    Traitement 2  
...  
else :  
    Traitement n
```







## Code python

```
Avant=True
```

```
while ( ..... ):
```

```
.....  
if ( ..... == "R"):
```

```
.....  
    Avant=False
```

```
elif (detector_feu( )== ..... ):
```

```
.....  
    Avant=.....
```





## Defi 1

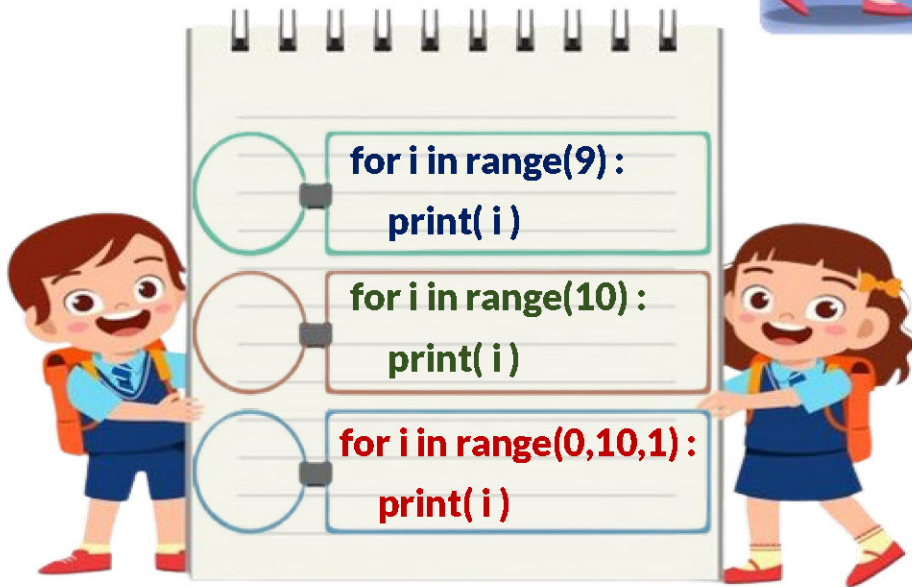
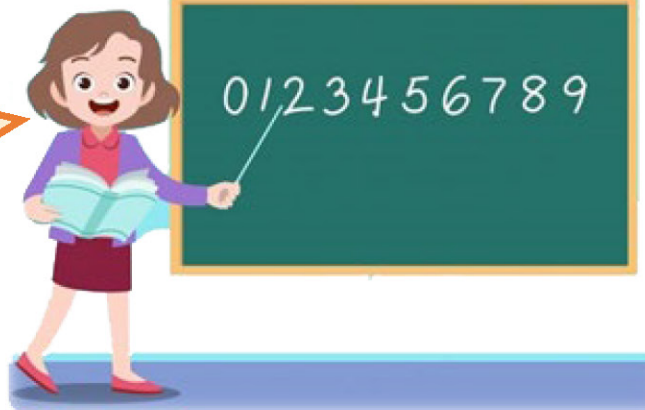
Pour tester si un entier X  
est positif on utilise :



|                       |       |
|-----------------------|-------|
| <input type="radio"/> | for   |
| <input type="radio"/> | if    |
| <input type="radio"/> | while |

## Defi 2

Mettre **vrai** ou **faux**  
dans chaque case, afin  
d'afficher les valeurs  
dans le tableau.



## Defi 3

Encercler la bonne  
réponse



Soit le script python suivant :

```
x=1  
while (x<=10):  
    x=x*2  
print(x)
```

La valeur de X est ??





# Robotique

MicroPython



ESP32



## LES ROBOTS

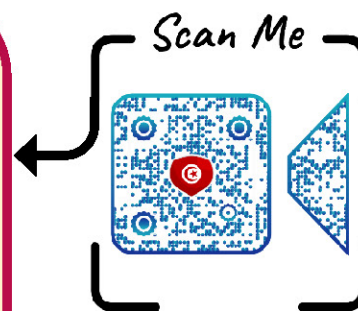
Scanner le QR code afin de visualiser la **CAPSULE N°1** pour une meilleure explication.



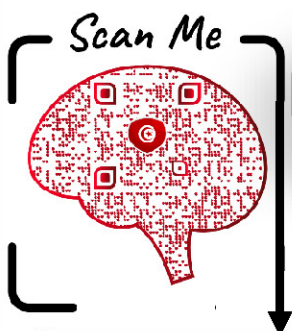
Les robots sont nombreux depuis les années 1970.

Ils exécutent des tâches répétitives avec une précision constante.

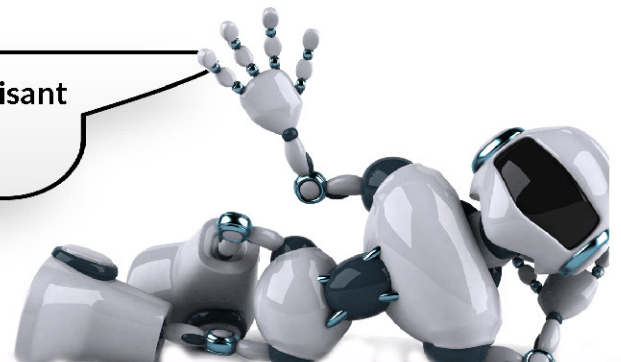
Avec les progrès de l'électronique et de l'informatique les robots sont de plus en plus complexes.



Scanner le QR code afin de visualiser la **CAPSULE N°2** pour une meilleure explication.



Définir le terme **ROBOT** en utilisant les termes au-dessous.



Programmable, Fonctions, Interpréter, Machine, Informations, Interagir, Automatique

.....

.....

.....

.....

# Les composants d'un robot



## Les capteurs

Leurs rôles sont de détecter des faits physiques (présence d'un objet, présence d'une chaleur, présence d'une lumière...).



## Le contrôleur

C'est le cerveau du robot, qui permet d'analyser les données provenant des capteurs et d'envoyer les ordres aux actionneurs (LED, Moteurs...).



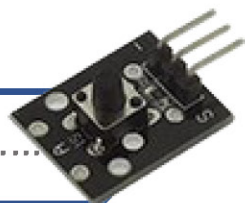
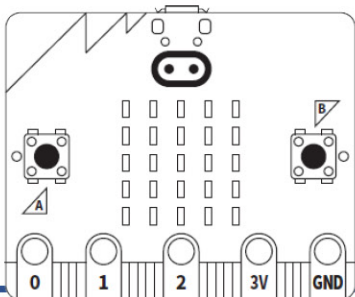
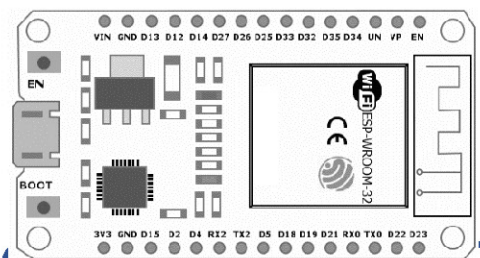
## Les actionneurs

Permettront au robot d'effectuer les actions commandées par son cerveau, en fonction du traitement des informations recueillies par les capteurs.

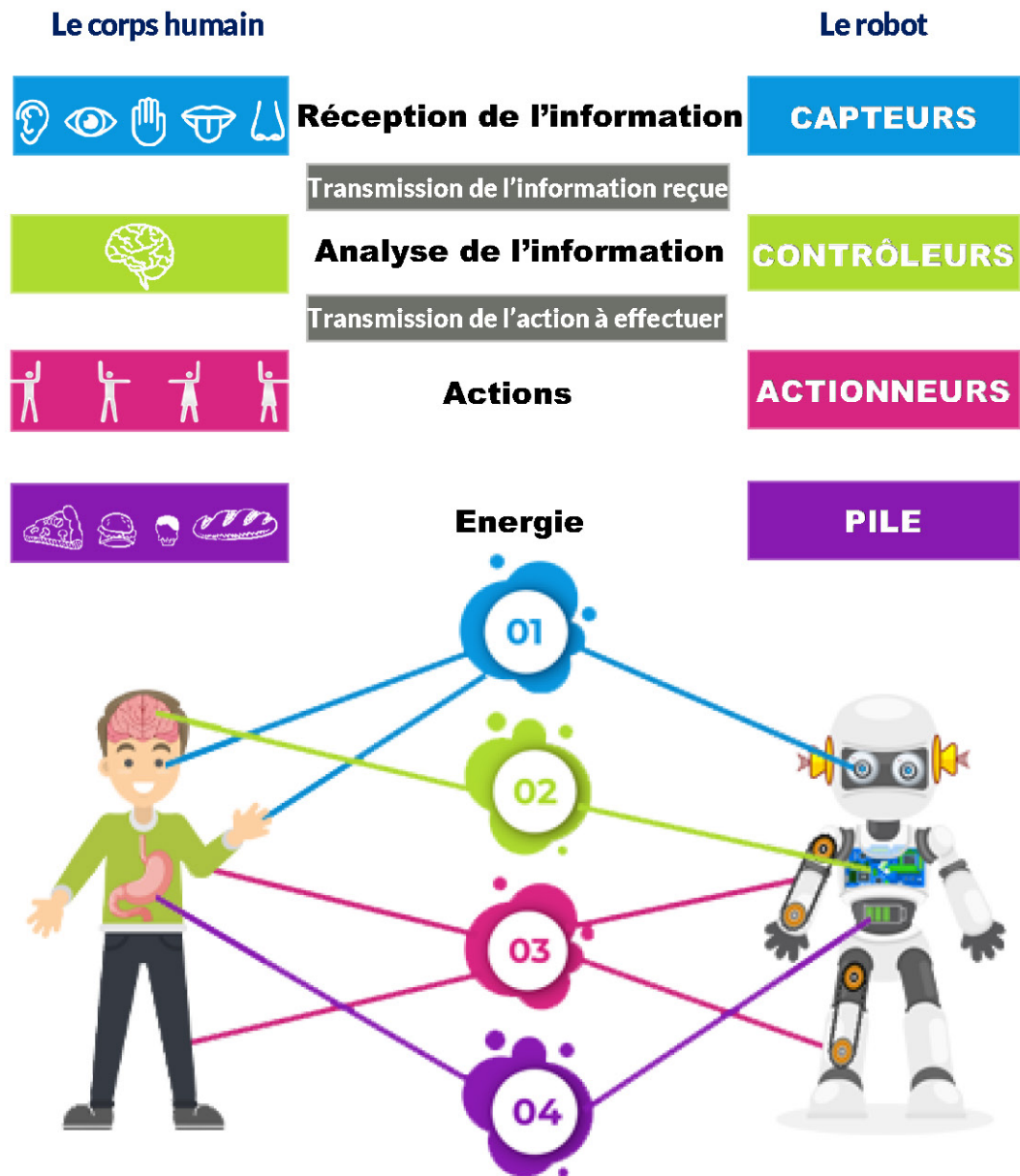


## A la découverte des composants

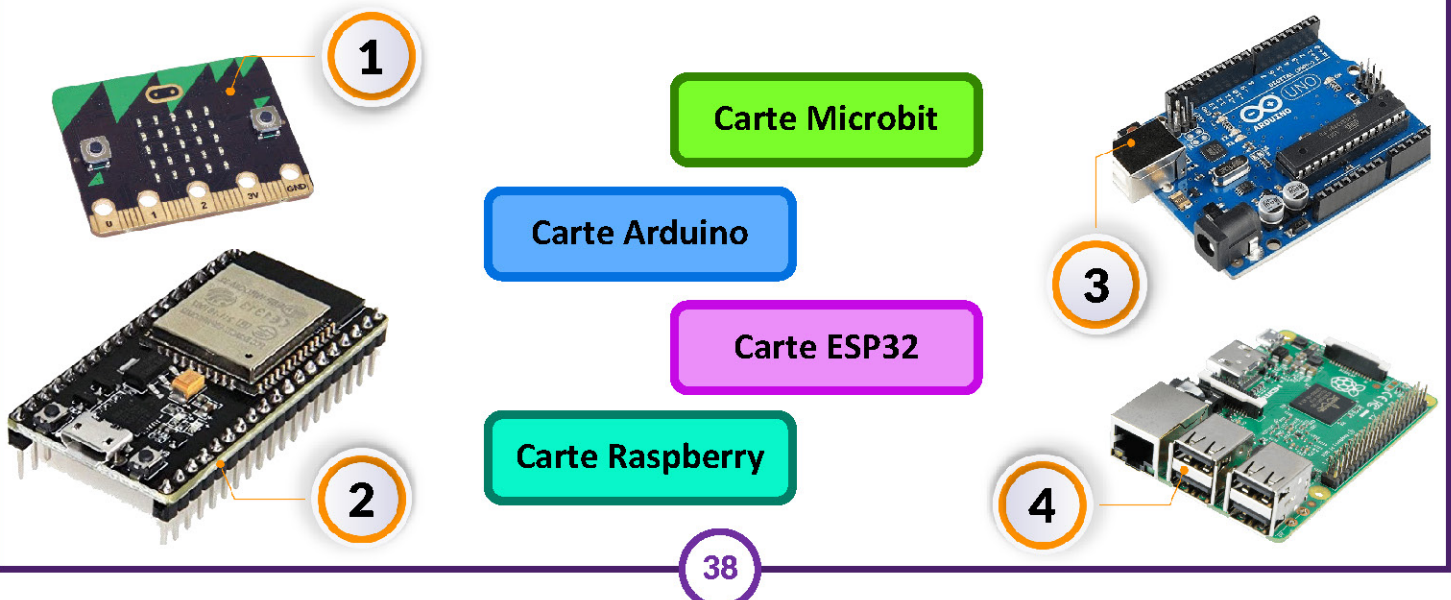
Identifier les actionneurs, les capteurs et les contrôleurs.



# LES POINTS COMMUNS ENTRE L'HUMAIN ET LE ROBOT



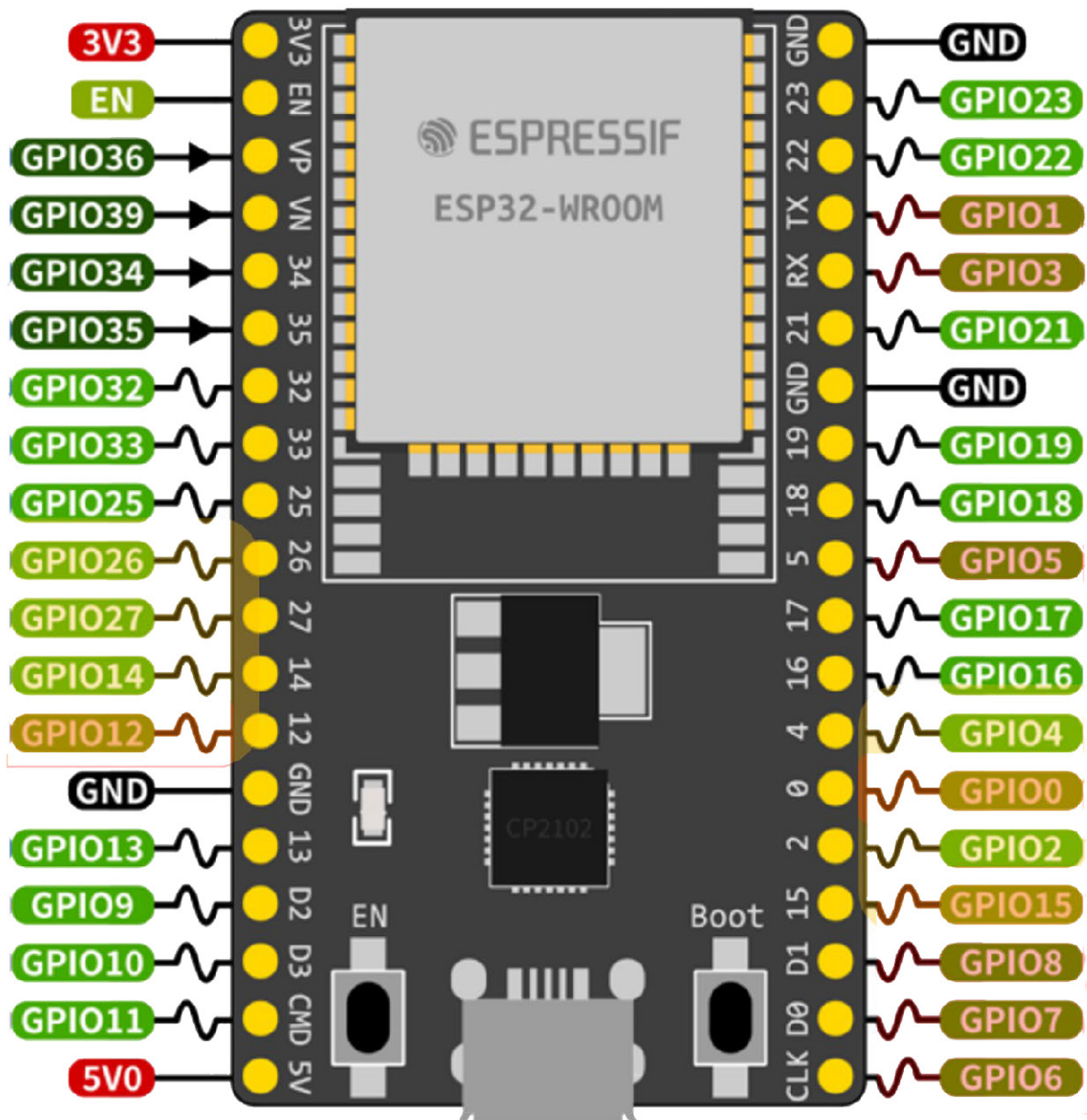
**RELIER CHAQUE CONTRÔLEUR (CARTE PROGRAMMABLE) AVEC SON NOM.**





## Présentation de la carte ESP 32

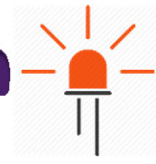
La carte ESP32 développée par la société ESPRESSIF, est une carte de développement à faible coût dédiée à l'internet des objets (IoT) et les applications embarquées. Elle est dotée d'une communication sans fil Wifi et Bluetooth.





# Robotique

MicroPython



ESP32



## FEUX DE CIRCULATION

01

Clignoter une diode LED



Accéder au site

<https://wokwi.com/>

Créer un nouveau projet :

MicroPython on ESP32

Une fois le projet crée :

- Mettre en commentaire la première ligne de code affichée par défaut.

- Cliquer sur le bouton PLAY

Une LED va s'allumer

Quelle est sa couleur ? .....

Que représente cette LED ? .....

.....



Ajouter au montage le code suivant, et l'exécuter

```
from machine import Pin  
led=Pin(2, Pin.OUT)  
led.on()
```

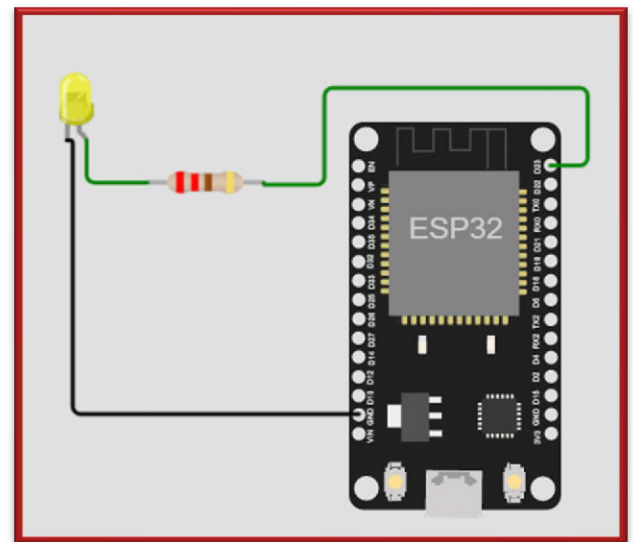
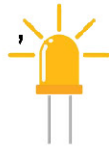


Une Deuxième LED s'allume

Quelle est sa couleur , et que représente cette LED ?

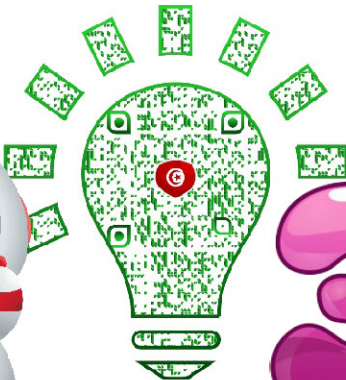


Ajouter une LED jaune ,  
branchée à la pin 23, et  
une résistance  
au montage précédent afin  
d'obtenir le branchement  
à droite.



Réexécuter le code précédent

Interpréter le résultat :



Remplacer **Pin(2, Pin.OUT)** par **Pin(23, Pin.OUT)**.

Déduire l'impact après exécution du code.







Modifier le code pour allumer la **LED**  
pendant **1 seconde** , puis l'éteindre pendant  
une **seconde** en utilisant :

 **led.off()** pour éteindre la **LED**

 **sleep(1)** pour attendre 1 seconde



Modifier le code pour  
faire clignoter la **LED**  
**5 fois de suite** .  
Tester le code .



Ecrire  
le  
**Code**



.....

.....

.....

.....

.....

.....

.....

.....

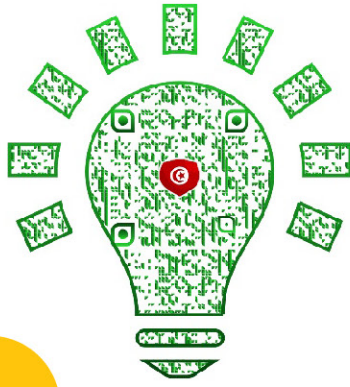
.....

.....

.....



**AJOUTER LA BIBLIOTHÈQUE **time** AU DÉBUT DU CODE POUR POUVOIR  
UTILISER LA FONCTION : **sleep**(valeur d'attente en secondes )**



Modifier le code pour  
faire clignoter la **LED**  
d'une manière  
continue , et le  
tester .



A large sheet of paper with horizontal lines, held by two pieces of tape, intended for writing code.

Implémenter, le code  
qui permet de faire  
clignoter la **LED**  
de manière  
continue

Monter le circuit  
approprié en utilisant  
les composants de la  
boîte fournie  
en classe

Enregistrer  
le code sous le  
nom : **Blink .py**

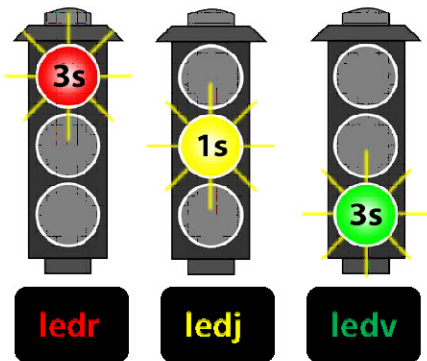
Exécuter  
le code enregistré



Par  
**Groupe**

02

## Programmer un Feu de circulation



ledv est branchée à pin 25 et clignote 3 fois pendant 3 secondes.

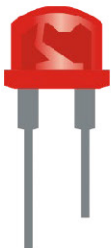
ledj est branchée à pin 32 et clignote 1 fois pendant 1 seconde.

ledr est branchée à pin 33 et clignote 3 fois pendant 3 secondes.



Compléter l'illustration du schéma permettant la réalisation du montage de la simulation des feux de circulation.

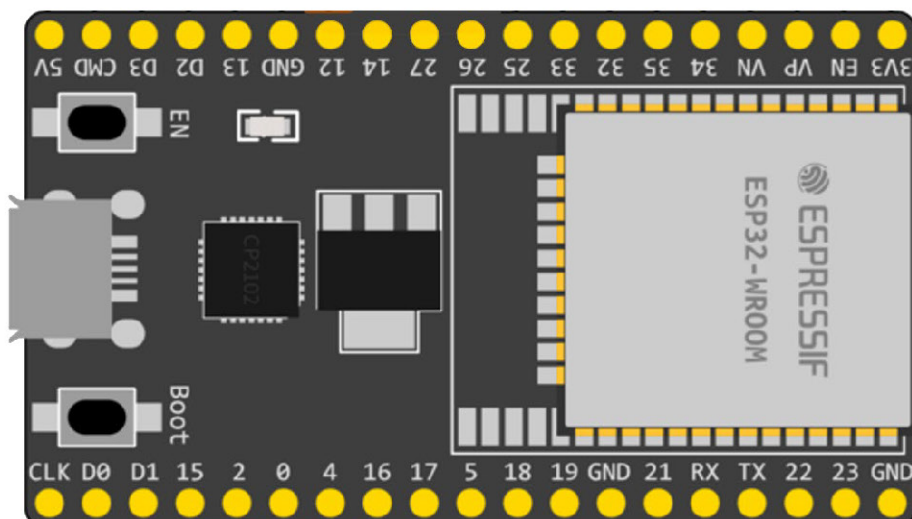
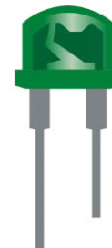
ledr



ledj



ledv





Écrire le code permettant de simuler le feu de circulation à l'aide de l'IDE (INTEGRATED DEVELOPMENT ENVIRONMENT), installé sur votre machine,

Enregistrer dans votre dossier de travail , le code sous le nom **feu.py** et le tester.



Area for writing code, featuring horizontal dotted lines.

Ajouter les **LED** aux  
feux présents dans  
votre maquette  
« Smart Cross  
Road »

Monter le circuit  
approprié en utilisant  
les composants de la  
boîte fournie  
en classe

Enregistrer  
le code sous le  
nom : **Feux.py**

Exécuter  
le code enregistré

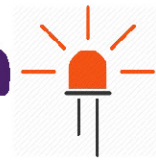
Par  
**Groupe**





# Robotique

MicroPython



ESP32



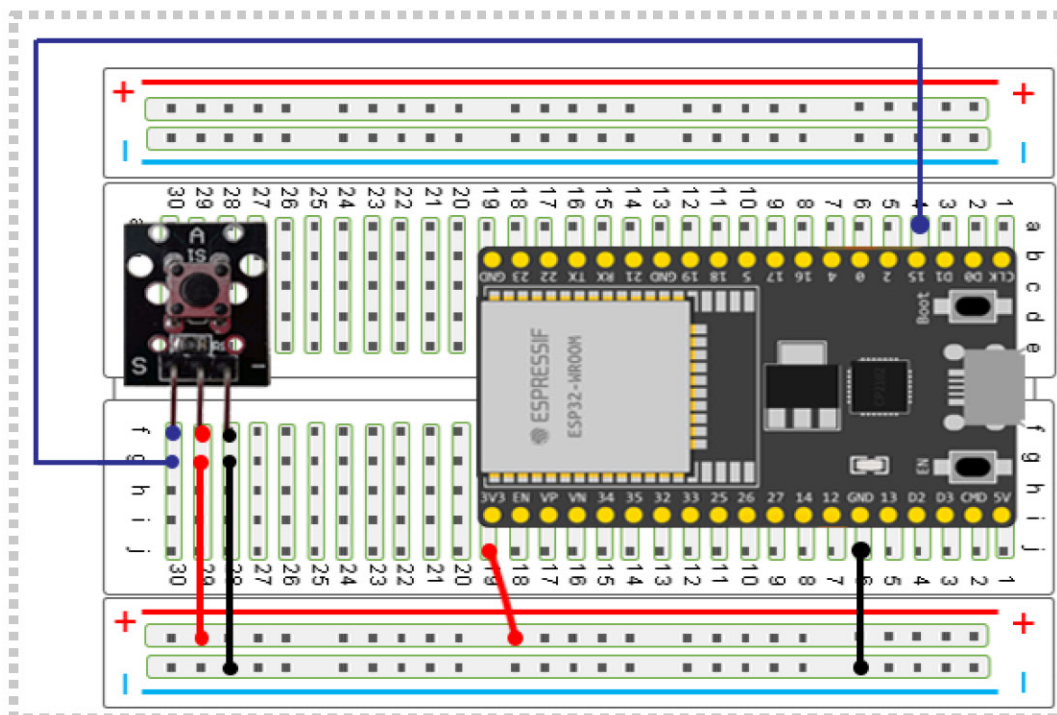
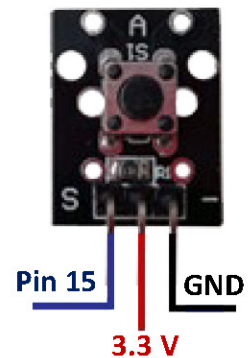
## FEUX PIÉTONS

01

Allumer une diode **LED** à l'aide d'un **bouton-poussoir**

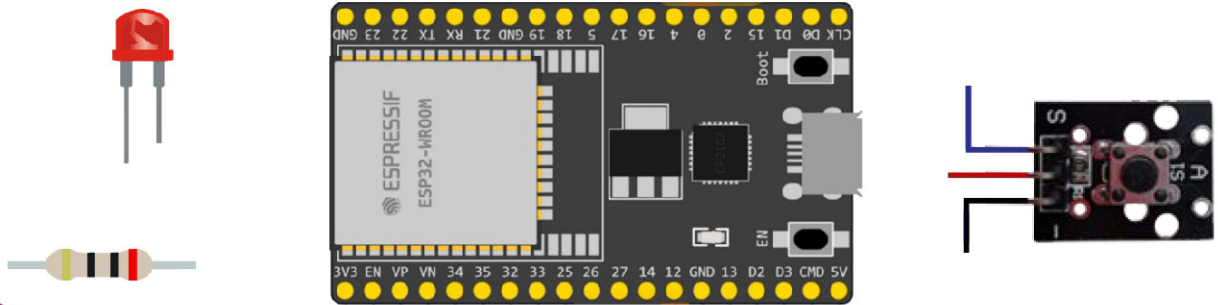


Créer le montage ci-dessous, en utilisant une plaque d'essai et le module du **bouton poussoir** présenté à droite.





Connecter la diode **LED** à la carte pour qu'elle s'allume à l'aide du **bouton -poussoir**.



Écrire et tester le code python permettant de faire allumer la **LED** si le **bouton poussoir** est enfoncé.

Ecrire  
le  
Code

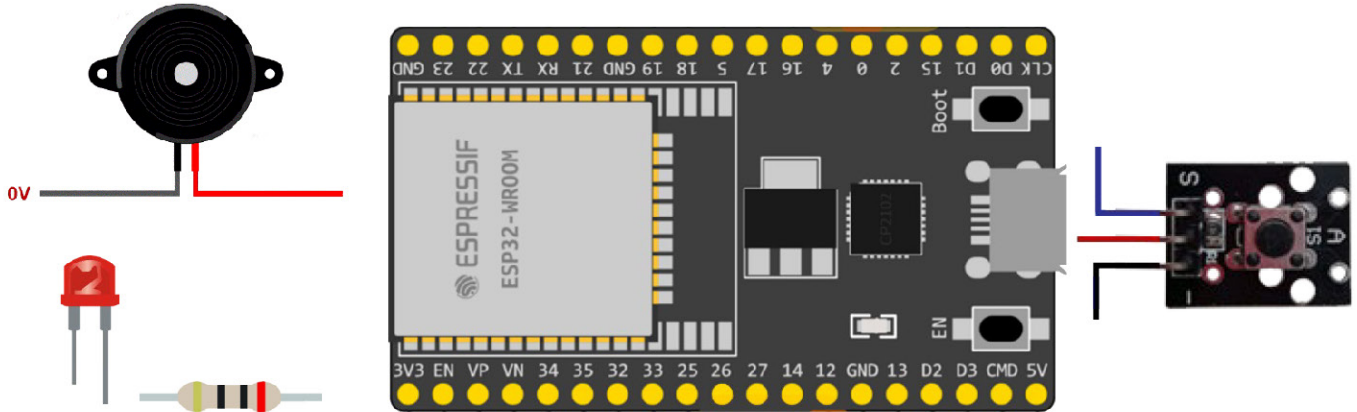


Blank lined area for writing code.

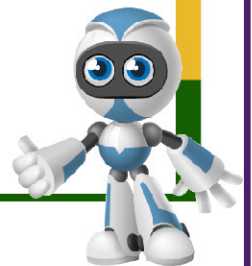


L'INSTRUCTION `boutonP.value( )` RETOURNE LA VALEUR 1 SI ON APPUIE SUR LE BOUTON, ET RETOURNE LA VALEUR 0 SI ON LE RELÂCHE.

Ajouter un **buzzer** à votre montage pour qu'il se déclenche au moment de l'allumage de la **LED** une fois le **bouton poussoir** enfoncé .



Réécrire le code, en ajoutant les instructions nécessaires afin d'activer le **buzzer** et allumer la **LED** au moment de l'appui sur le **bouton poussoir** . Tester le code.



## Rassembler vos acquis, et finaliser le projet

### Smart Cross Road



Le projet consiste à utiliser les prérequis pour réaliser un feu de circulation permettant à la fois de réguler la circulation et de permettre aux piétons d'avoir la main pour traverser en toute sécurité. Je vais vous guider à travers des conseils et astuces afin que vous puissiez faire les premiers pas vers la réalisation de votre Smart Cross Road.

**Importation  
des  
bibliothèques**



Broches des LED pour le feu de circulation, à écrire au début du code.

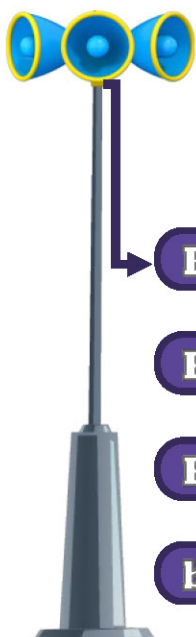
```
from machine import *
```

```
from time import sleep
```

```
ledj = Pin (32 , Pin.OUT)
```

```
ledr = Pin (33 , Pin.OUT)
```

```
ledv = Pin (25 , Pin.OUT)
```



```
Buzz = Pin (19 , Pin.OUT)
```

```
PledV = Pin (23 , Pin.OUT)
```

```
PledR = Pin (22 , Pin.OUT)
```

```
boutonP = Pin (21 , Pin.IN)
```



Broches des **LED** pour le feu des piétons, ainsi que le buzzer et le bouton-poussoir.



Tiens, tiens,  
il manque une valeur au  
niveau de la condition de  
marche. Compléter la  
pour que le code  
tourne correctement.

### BLOC FEUX DES VOITURES

**while** boutonP.value( ) == ☐

ledv.value(1)

sleep (1)

ledv.value(0)

ledj.value(1)

sleep (1)

ledj.value(0)

ledr.value(1)

sleep (1)

ledr.value(0)

PAS DE  
PIÉTON

Les feux, VERT (ledv) , JAUNE (ledj) et ROUGE (ledr)  
s'allument successivement pendant une seconde.  
Ce cycle se répète en continue tant qu'il n'y a pas un  
piéton qui compte traverser la rue.

### BLOC FEUX DES PIÉTONS

**if** boutonP.value( ) == ☐

Buzz.value(☐)

PledV.value(1)

PledR.value(0)

ledv.value(0)

ledj.value(0)

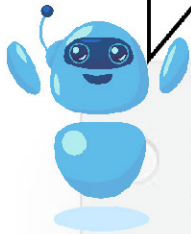
ledr.value(1)

Voilà un piéton qui demande à  
traverser la rue en appuyant  
sur le bouton rouge.

Le buzzer doit se déclencher,  
le feu rouge s'allume pour les  
voitures, et le feu piétons

bascule du  
rouge  
au vert

Youppie ! Maintenant que tout est prêt il ne reste qu'assembler les différents blocs au sein d'un même code qui tourne à l'infinie. N'oublier pas que si le piéton appuie sur le bouton, le feu piéton passe au vert et celui pour les voitures passe au rouge. Dans le cas contraire, le feu de circulation pour les voitures commence à tourner (Vert -> Jaune -> Rouge) et le feu rouge pour les piétons est allumé



```
from machine import *
```

```
from time import sleep
```

```
ledr = Pin( , Pin.OUT)
```

```
ledj = Pin( , Pin. )
```

```
ledv = Pin( , Pin. )
```

```
Buzz = Pin( , Pin. )
```

```
PledV = Pin( , Pin.OUT)
```

```
PledR = Pin( , Pin.OUT)
```

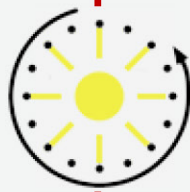
```
boutonP = Pin( , Pin. )
```

```
while True :
```

```
    if boutonP.value() == 1 :
```

```
    else:
```

CLIGNOTEMENT  
3 SECONDES



```
while boutonP.value() == 0:
```

```
    ledv.value( 1 )
```

```
    sleep( ... )
```

```
    ledv.value( ... )
```

```
    for i in range( ... ):
```

```
        ledj.value( ... )
```

```
        sleep( ... )
```

```
        ...value( 0 )
```

```
        sleep( 0.5 )
```

```
    ledr.value( ... )
```

```
    sleep( ... )
```

```
    ...value( 0 )
```

Optionnel

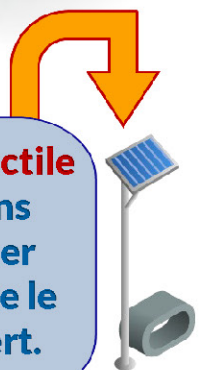


Améliorer votre code pour :

**1** Faire clignoter 3 fois les feux, rouge et vert (du feu circulation) avant d'être éteints.

**2**

Ajouter une **touche tactile** permettant aux piétons non-voyants de signaler leur présence, pour que le feu piéton passe au vert.





03

## À la découverte des Capteurs tactiles

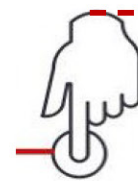


Observer l'annexe de la carte ESP32 et repérer les 10 capteurs tactiles nommés **TOUCH**

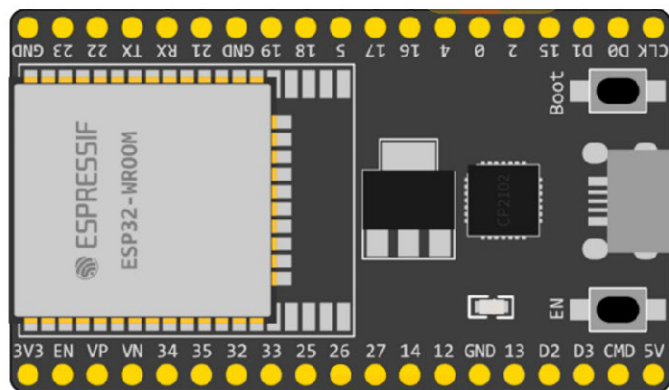
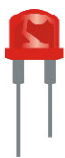
### RÉFÉRENCE CAPTEURS TACTILES DE LA CARTE ESP32



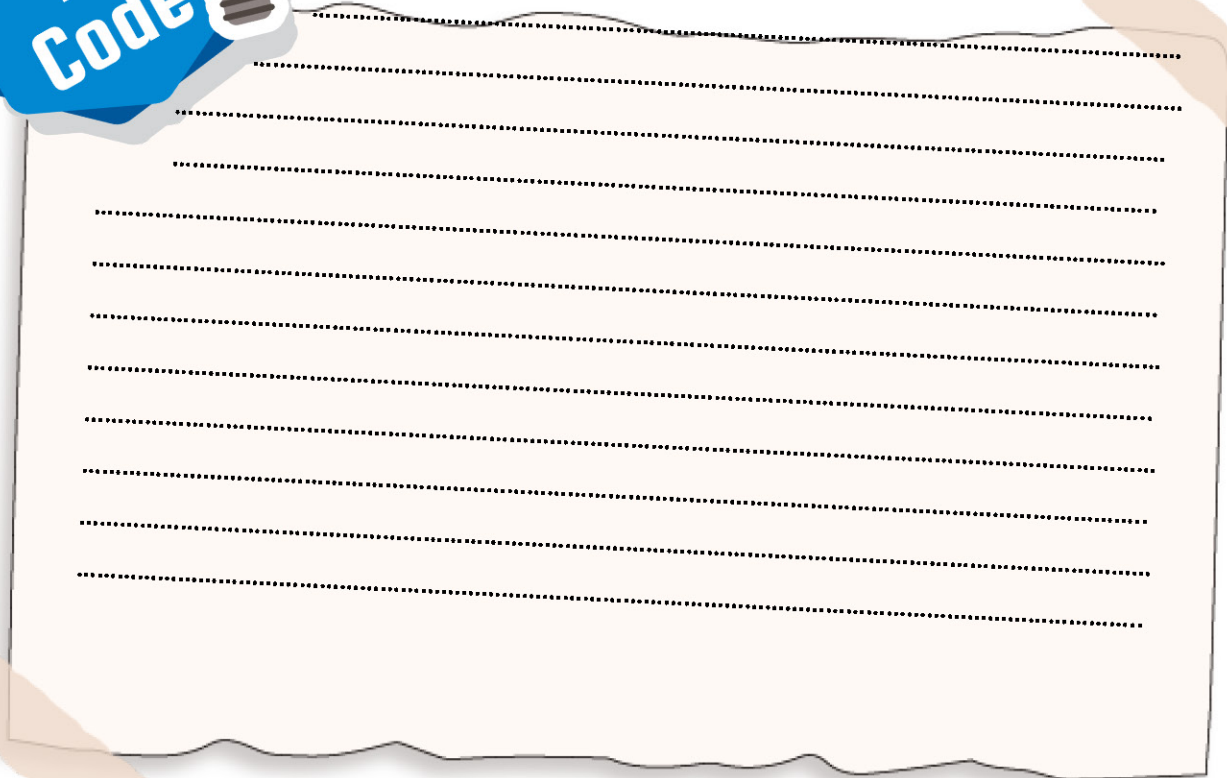
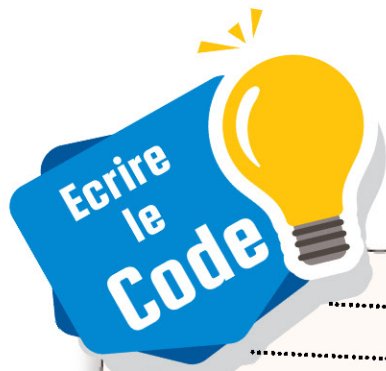
Compléter le montage afin d'allumer la **LED** à l'aide d'un **CAPTEUR TACTILE**.



monTouch



Écrire et tester le code python permettant de faire allumer la **LED** si le **CAPTEUR TACTILE** détecte une touche humaine.

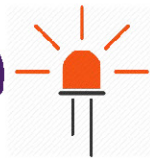


**L'INSTRUCTION `Touch0.read( )` LIT LA VALEUR DE LA PIN TACTILE**

**L'INSTRUCTION `Touch0.config(sensibilité)` DÉFINIT LE SEUIL D'ARRÊT DE LA TOUCHE TACTILE.**



MicroPython



ESP32



# Robotique

## ANNEXE

